A COMPUTER SOLUTION TO THE DAILY FLIGHT SCHEDULE PROBLEM

Craig Gibson Honour

THESIS

A COMPUTER SOLUTION TO THE

DAILY FLIGHT SCHEDULE PROBLEM

by

Craig Gibson Honour

June 1975

Thesis Advisor: U. R. Kodres

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

REPORT DOCUMENTATION PAGE		READ INSTRUCTIONS BEFORE COMPLETING FORM
1. REPORT NUMBER	2. GOVT ACCESSION NO.	3. RECIPIENT'S CATALOG NUMBER
4. TITLE (and Subtitle) A Computer Solution to the Daily Flight Schedule Problem		5. TYPE OF REPORT & PERIOD COVERED Master's Thesis; June 1975 6. PERFORMING ORG. REPORT NUMBER
7. AUTHOR(s) Craig Gibson Honour		8. CONTRACT OR GRANT NUMBER(s)
9. PERFORMING ORGANIZATION NAME AND ADDRESS Naval Postgraduate School Monterey, California 93940		10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS
11. CONTROLLING OFFICE NAME AND ADDRESS Naval Postgraduate School Monterey, California 93940		12. REPORT DATE June 1975 13. NUMBER OF PAGES 74
14. MONITORING AGENCY NAME & ADDRESS(if different from Controlling Office) Naval Postgraduate School Monterey, California 93940		15. SECURITY CLASS. (of this report) Unclassified 15a. DECLASSIFICATION/DOWNGRADING SCHEDULE

16. DISTRIBUTION STATEMENT (of this Report)

Approved for public release; distribution unlimited.

17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)

18. SUPPLEMENTARY NOTES

19. KEY WORDS (Continue on reverse side if necessary and identify by block number)

scheduling problem
computer
interactive
conflict graph
flight schedule
assignment problem

20. ABSTRACT (Continue on reverse side if necessary and identify by block number)

A computer procedure to solve interactively the daily flight scheduling problem for training squadrons is proposed. The scheduling problem for a prototype squadron, Fighter Squadron One Hundred Twenty-one, is mathematically analyzed using graph coloring techniques. A procedure similar to published class scheduling solutions which uses an assignment algorithm is formulated. A computer program is then developed to demonstrate the procedure.

DD FORM 1 JAN 73 1473 EDITION OF 1 NOV 65 IS OBSOLETE
S/N 0102-014-6601 |
(Page 1)

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

A COMPUTER SOLUTION TO THE DAILY FLIGHT SCHEDULE PROBLEM

by

CRAIG GIBSON HONOUR
LIEUTENANT , UNITED STATES NAVY
B. S., UNITED STATES NAVAL ACADEMY, 1968

Submitted in partial fulfillment of the
requirements for the degree of

MASTER OF SCIENCE IN COMPUTER SCIENCE

from the

NAVAL POSTGRADUATE SCHOOL
June 1975

ABSTRACT

A computer procedure to solve interactively the daily flight scheduling problem for training squadrons is proposed. The scheduling problem for a prototype squadron, Fighter Squadron One Hundred Twenty-one, is mathematically analyzed using graph coloring techniques. A procedure similar to published class scheduling solutions which uses an assignment algorithm is formulated. A computer program is then developed to demonstrate the procedure.

TABLE OF CONTENTS

ACKNOWLEDGEMENT

The author would like to acknowledge the contributions of LT. Chuck Porter, LT. John Helm and LT. Bob Dickerson from Fighter Squadron One Hundred Twenty-one who generously provided the necessary information for analysis, Dr. Uno Kodres whose mathematical expertise was of invaluable assistance in analyzing this problem, LCDR Flack Logan who provided constant liason between Monterey and Miramar and the author's wife Sherry Honour who provided the encouragement and secretarial skills necessary for successful completion.

The algorithm discussed in this thesis [Reference 8] is republished after translation from ALGOL to COBOL. General permission to republish, but not for profit, has been granted by the Association for Computing Machinery.

I. INTRODUCTION

Rising personnel costs and declining computer costs indicate that a reassessment of many job functions now performed manually within the Navy is necessary. One such task, the daily flight schedule problem, which is essential to efficient squadron operation is still performed entirely by hand. Many commands recognize the demanding nature of this task and, in many instances, smooth operation of the scheduling task requires the talents of more than one officer.

It is the contention of this thesis that the student and instructor flight schedule produced by training squadrons can be obtained efficiently by one officer with the aid of a computer; this can result in significant personnel savings. A prototype squadron, Fighter Squadron One Hundred Twenty-one, was selected for which to develop a preliminary computer program in order to verify that the computer procedures result in acceptable solutions.

The first section of this paper describes the prototype problem. Included are discussions of the variables, current scheduling techniques and desired solution parameters. The next section describes the mathematical analysis used to arrive at a solution technique. Two techniques for solving class scheduling problems are presented with the aid of an example. The next section contains a description of the computer program designed to implement the solution. The computer program is listed in the appendix. Results in the form of actual printout from a sample run, using data

collected from flight schedules, are also presented. The last section is the proposal for an extension to an interactive system which presents hardware requirements and software modifications.

II. THE SCHEDULING PROBLEM

A. GENERAL BACKGROUND

Fighter Squadron One Hundred Twenty-one (VF-121) is tasked to provide combat ready fleet replacement pilots and radar intercept officers (RIO's) to fly the supersonic F4J Phantom II aircraft. Located at NAS Miramar, California, it has assigned a semi-permanent staff of fleet experienced pilots and RIO's who instruct student pilots and RIO's in the fundamentals of the F4J operation.

Long range training goal decisions based on asset availabilities, progress, anticipated delays and fleet requirements are reflected through command directives to the scheduling office. The scheduling staff analyzes the variables and produces a daily flight schedule to fulfill the training goals. An optimum flight schedule is critical for efficient squadron operation. The working hours of maintenance personnel are adjusted daily to meet anticipated needs. Outside agencies such as the Federal Aviation Agency are alerted by the schedule to provide military and civilian aircraft separation. Weapons ranges use the schedule for manning requirements. Support agencies operating cockpit trainers and related facilities are alerted for scheduled usage. Consequently, the daily flight schedule not only regiments squadron personnel but also provides valuable information to a variety of agencies.

B. PROBLEM VARIABLES

1. Instructors

Both pilots and radar intercept officers are assigned as instructors for a three year tour. They progress through a formal qualification syllabus named the Instructor Under Training (IUT) syllabus. As an instructor completes a phase he becomes qualified to teach that phase. Therefore, at any given time, instructors will be only qualified for certain events. The maximum number of instructor pilots or RIO's is fifty each. They are divided into six wings for rotation purposes. A particular wing will have night duty for a week and rotate to day flying for the next three weeks. The last two wings consist of special category instructors such as those qualified but not assigned to the squadron and the Landing Signal Officer (LSO) wing whose additional night duties merit special attention.

Command regulations limit instructors to a ten hour working day to preclude excessive fatigue, although some instructors are further limited to one event per day for other reasons. Instructor scheduling should distribute flight time evenly, including night and weekend flying, and insure that all minimum flight time requirements are satisfied.

2. Students

Student pilots and RIO's are categorized by previous experience and anticipated needs of the Navy with one basic

syllabus serving all categories. However, deletion of nonessential events for the more experienced categories is a standard procedure. Students range from Ensigns, who have just completed flight training, to senior Commanders, who have been selected as Carrier Air Group Commanders. Students are divided into classes with a maximum of thirty pilots or thirty RIO's for event scheduling at any one time. Each class passes through a Prior To Flight phase (PTF) which is scheduled through a different office. PTF training is received as a block and is annotated on the back of the flight schedule. It causes no conflicts since a student is not released from PTF until its completion.

Students are limited to an eight hour working day. Classes should progress through the syllabus evenly to provide maximum lecture opportunities and effective aircraft utilization for joint missions. Some students or even entire classes may receive priority. Generally, different classes will be in different phases signifying various stages of syllabus completion. A student pilot will normally require the services of an instructor RIO for a sortie and a student RIO will normally be paired with an instructor pilot. Other combinations are occasionally scheduled except two RIO's can never fly together.

3. Events

Scheduled events may be divided into four groups: sorties, lectures, trainers and special events. The first, student training sorties, are listed in the syllabus and are scheduled essentially consecutively. A student will progress through the four major phases of familiarization, radar training, conventional weapons delivery and tactics so that successful completion of the current flight is a prerequisite to the next flight. An incomplete or failing

grade will normally result in at least one re-fly of the same sortie. A student may begin the next phase when only a few sorties remain in the current phase and thus be eligible for two different types of sorties at the same time.

The maximum number of flights for either a student pilot or RIO is fifty. Asset complexity ranges from one aircraft with no special configuration for a basic familiarization flight to a specially configured section (two aircraft) with an adversary section (dissimilar aircraft), a special Air Combat Maneuvering Range (ACMR), separate frequencies and appropriate weather minimums. Some flights require successful completion of another sortie the same day or within one day. Others, such as Field Carrier Landing Practice, may require a separate instructor sortie earlier in the day to transport the LSO to a remote location so that he will be prepared to function as an LSO. Many flights are restricted to daytime hours while others are strictly night flights. Some may be flown either time. Most requirements for an individual flight are described in the VF-121 syllabus course description but current scheduling practices and techniques are not completely delineated on paper.

The second major group of events is the student lecture syllabus. There is a maximum of sixty lectures which are the same for student pilots, RIO's and IUT's. Lectures are prequisites for certain flights or phases and are scheduled to be completed no earlier than two weeks prior to that flight. Several classrooms are available, but only certain instructors may give each lecture. Lectures are not scheduled for individuals but for entire classes including IUT's. Normally, one student being unavailable will preclude assignment of a lecture for that time.

Ground based training aids are the third major group

of scheduled events. There are three types of F4 trainers which are shared by all the F4 squadrons at Miramar. The emergency procedures trainer requires only one student. The cockpit mockup trainer requires a pilot, RIO and an instructor. The radar systems trainer requires one student and an instructor. All trainers are scheduled for hourly periods with the VF-121 scheduling staff coordinating all squadron times. Training periods may result in incomplete training which necessitates rescheduling. The trainer syllabus is designed to give practical experience to classroom techniques.

The last major group consists of special events which must be scheduled. These may consist of a test flight requiring a test qualified instructor pilot and RIO or a cross country ferry flight. Ground events such as All Officer Meetings directed by higher authority could also be included in this category. Any event not in the syllabus but which must be indicated on the daily flight schedule is a special event. Usually required crews or personnel are already determined but sometimes the scheduling staff must provide the appropriately qualified personnel.

4. Assets

The scheduling staff receives from the maintenance department an estimate of the number of available aircraft for the next day. An attempt is made to fully utilize assets but not to overschedule. Aircraft assignments are made by the maintenance department approximately two hours before scheduled launch time to insure that the correct configuration is available.

Range availability is requested and received monthly along with various frequency assignments. Military

intercept controllers are available daily and are assigned to coincide with the published flight schedule. Weather forecasts are available from the station weather facility. Marginal forecasts often require contingency scheduling since weather minimums vary for different flights.

5. Carrier Qualification

A major uncontrolled variable in scheduling parameters is aircraft carrier qualification, which is normally the culmination of a student's training. Since the asset demands are so severe, all assets are programmed to insure maximum utilization of the infrequently assigned aircraft carrier deck space. Classes are slowed or expedited to be ready for certain carrier assignments. During carrier qualifications, most available aircraft are used for qualification with some spares held in reserve. Any aircraft remaining may be utilized for normal syllabus training. For several weeks preceeding qualifications a maximum number of aircraft are used for night carrier field landing practice, often during early morning hours or on weekends. The scheduling staff receives crew assignments from the LSO for these night flights and must adjust the day schedule accordingly. The result is usually a shortage of qualified instructor RIO's and difficult scheduling conflicts.

6. Personnel Nonavailability

The last major variable is aircrew nonavailability. Many causes for nonavailability exist but all present the same problem. Personnel nonavailability may range from a fifteen minute dental check-up to a medical "down" for a broken leg. The scheduling staff receives all programmed

nonavailability, such as watch bills and schools, and maintains a "snivel" log up to a week in advance for individual anticipated absences. Prior to scheduling, the staff must insure aircrew availability.

C. CURRENT SCHEDULING METHOD

Three officers and an enlisted typist are assigned to the staff. The two senior officers rotate the scheduling duties to enable one of them to be available for flying duties. The third officer acts in an information gathering role and the typist types and reproduces the schedule. The next days schedule is prepared during the previous day and is usually completed by 1800 hours. Actual launch times are staggered to accomodate limited launch assets. The first three cycles are day launches and the last is a night launch.

Normally, the duty scheduler reports to work early and evaluates the information found on the Squadron Duty Officer's smooth schedule giving completed sorties and flight times. This information is transcribed to a set of "grease boards" which gives current status of all students and instructors. The next step is to line out all nonavailable personnel on the grease board schedule and enter basic information such as duty officers, date, sunrise and sunset times. This entire process consumes about two hours.

The next process varies with the individual in charge but is usually the scheduling of high priority events such as: sorties to use fixed range times or special events. Included here are lectures or trainers which, if not scheduled, will create student nonavalability within two

days. Liason with holders of external assets (such as dissimilar aircraft) is established to find mutually agreeable sortie times for the tactics phase flights. Other missions including IUT sorties are then scheduled to utilize the rest of available aircraft for each cycle. Sorties are only scheduled by mission and student. The scheduler anticipates opportunties for scheduling entire classes for low priority lectures, possible exhaustion of qualified instructors and poor weather conditions. Often some classes will be given priority to enable sooner completion dates. This entire time consuming process usually requires about four hours to complete.

The next major step is to assign qualified instructors to all necessary flights, lectures and trainers. In addition to avoiding conflicts, care must be exercised to evenly distribute monthly flight time, maintain appropriate minimums and rotate undesirable flights. Depending on schedule construction to this point, there may or may not be a solution. No solution will necessitate a chain of modifications throughout the schedule, replacing missions, students and instructors. In its worst form the whole schedule may be scrapped and started over. Successful completion of this stage can usually be accomplished in two hours.

At this point the schedule is meticulously checked for unresolved conflicts. Any remaining unscheduled assets, such as trainers or classrooms, are attempted to be utilized. All additional information (such as notes and positive control routes and numbers) are filled in and the typist transcribes the grease board schedule onto a duplicating mat. The scheduler rechecks for errors and submits the mat for the Operations Officers signature. The daily flight schedule is then reproduced and distributed.

D. DESIRED SOLUTION PARAMETERS

User oriented discussions identified seven major areas of concern for any computer solution which must have a reasonable assurance of satisfication before such a procedure would be used in practice.

1. RESPONSIVENESS TO THE OPERATOR - The system must provide the operator sufficient decision control including a full manual input system to enable the operator to exert positive control over the solution.

2. BACKUP TRACE - The system must periodically dump files on paper to provide a backup manual method in the event of a massive system failure.

3. LOW MAINTENANCE REQUIREMENT - Both hardware and software maintenance requirements should be minimal since no trained personnel are available within the command.

4. OPTIMUM - The solution should produce an optimum daily schedule but be responsive to long range requirements.

5. EASY OPERATION - Since operators have no computer experience, the system should be easy to operate including file updating procedures. An operators handbook should be provided.

6. EXPANSION - The system should be capable of moderate expansion to provide improved management information, such as status charts or student flight time.

7. COBOL - The software must conform to NAVY ANSI standard COBOL to enable easy hardware procurement and software versatility.

Implementation would be gradual in that the user would have the opportunity to make reasonable changes and view test solutions prior to acceptance. The system should operate experimentally in conjunction with present methods for at least a month to determine operating restrictions, insure positive debugging and assess long range scheduling ramifications.

III. THE MATHEMATICAL ANALYSIS

A. PREVIEW

The key ingredient to a successful mathematical analysis of any problem is to extract the essential information and relate it to known concepts. The flight scheduling problem lends itself well to this type of approach. The basic variables are in many respects similar to school class scheduling problems which were the subject of much attention during the sixties. School adminstrators used two different approaches for a solution. The first approach, which makes use of graph coloring techniques, produces a schedule so that the student's wishes and requirements are satisfied. The other major method, the fixed schedule method, centers priority on the university requirements. A course offering is designed so that classrooms and professors are not in conflict. The students register for the courses which are offered at the indicated times. The students may not be able to take all the courses they desire.

The flight scheduling problem has features which are identical to the school scheduling problem. The procedure this paper develops to solve the problem is similar to the fixed schedule method which uses priorities. The students who cannot take the courses because of conflicts are given a higher priority for the next day's schedule.

B. COMPARISON TO SCHOOL TIMETABLE PROBLEMS

School timetables are constructed based on the following concepts. A set of professors (P) exist who are qualified to teach certain classes. Usually, a list is submitted to the registrar giving instructor names and the sections they will teach. A list of students (S), each of whom has indicated his desired courses (C), also exists. Often there are many other restrictions to complicate the basic idea. Courses are limited in size and may require either specific classrooms or excessive time slices such as laboratories. Classroom facilities may be limited. Instructor or student schedules must not exceed a certain prespecified number of hours. Certain periods must be reserved for meals. Also, there is an inherent system of prerequisites which requires that certain courses must be completed or scheduled concurrently with other courses.

The aircrew scheduling problem has the same critical variables. Student pilots and RIO's (S) must take certain events (C) taught by qualified instructors (P). The distinguishing factors are the restrictions. Instead of a course segment size of say thirty students, the aircrew problem is limited to one, except for lectures which are identical. Aircraft assets are limited in the same way as classroom facilities. Time limits and special requirements are different but are common to both problems.

C. GRAPH COLORING APPROACH

This section informally presents the basic concepts of

graph theory that are necessary to understand the graph coloring approach to solving scheduling problems. For a more detailed presentation the reader is invited to consult a standard textbook on graph theory such as one by Busacker and Saaty [1]. For use in this paper, a graph is a convenient visual representation of the scheduling problem and consists of a set of vertices (points) interconnected by a set of edges (lines).

The graph coloring technique for solving class scheduling problems is the most advanced and mathematically efficient solution. The classes are represented by vertices of a graph. Conflicts are depicted by connecting two vertices with an edge. The graph coloring problem is to color each vertex of the graph with some color so that if two vertices are connected by an edge then distinct colors appear on the vertices. This corresponds to saying that if two courses have a student who wishes to take them both, the courses must be scheduled at different times.

Course Number	Course Name	Professor
C1	M-101	P3
C2	H-203	P1
C3	S-211	P2
C4	S-105	P3
C5	E-207	P4

Figure 1. Courses Available.

An illustrative example consists of determining a nonconflict schedule for four professors who will teach a total of five classes as listed in Figure 1. Notice that the third professor teaches two separate classes. Six students must take different combinations of classes as listed in Figure 2.

Student	Courses
S1	C1,C2,C3
S2	C1,C2
S3	C1,C3,C4
S4	C2,C3
S5	C2,C3,C5
S6	C4,C5

Figure 2. Student Course Preferences.

Let C1 represent course one, C2 represent course two and so forth. Let Si represent the i-th student and Pj the j-th professor. The courses may be represented by vertices of a graph labelled C1 through C5 vertically. The set of students may also be represented by a vertex for each student labelled S1 through S6 and arranged vertically to the left of the courses. The professors are handled by the same technique and arranged vertically on the right of the courses. The three columns of vertices representing students, courses and instructors, as shown in Figure 3, is a tri-partite graph (G).

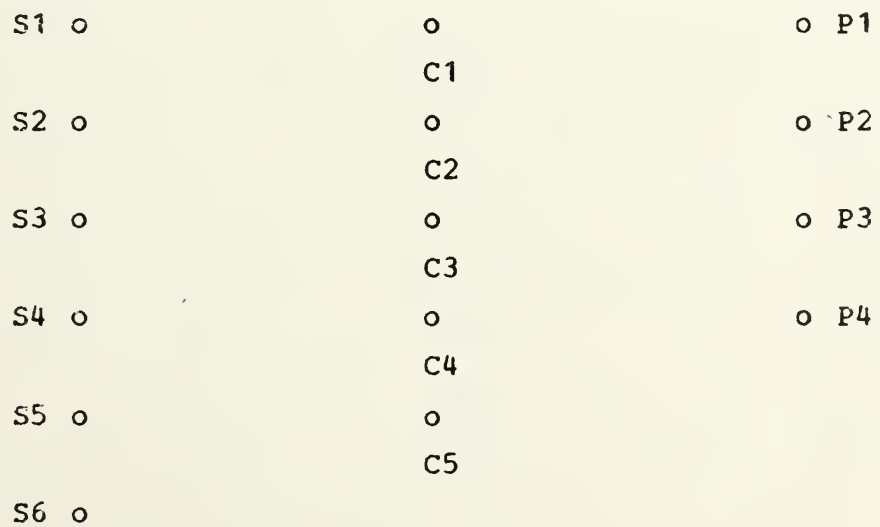

Figure 3. Vertices of a Tri-partite Graph.

Construct edges in graph G from vertex S1 to vertices C1, C2 and C3 to represent student S1 desiring courses C1, C2 and C3. Construct similar edges for the rest of the students. Now construct an edge from professor P1 to course C2 to represent professor P1 teaching course C2. Construct similar edges for the rest of the professors. The completed graph is shown in Figure 4.

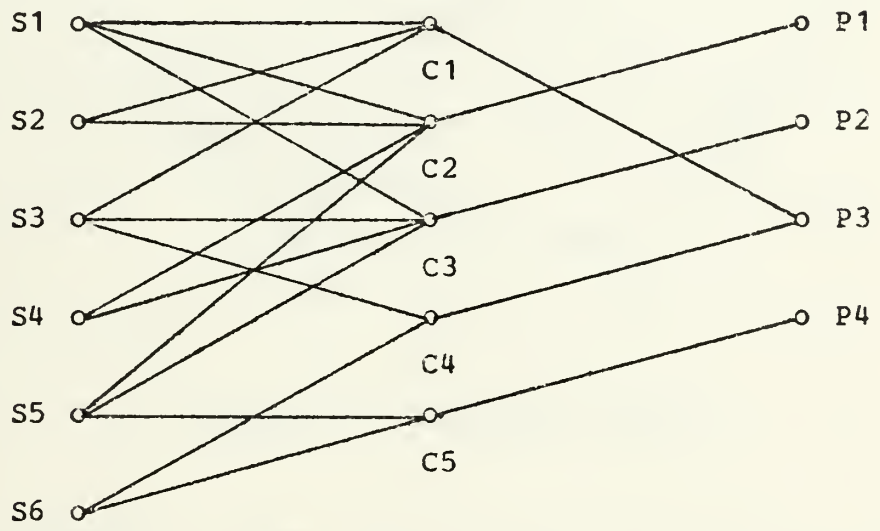

Figure 4. The Graph Depicting Schedule Requirements.

Now determine a conflict graph in the following manner. Construct a graph K with vertices to represent the courses offered. If in graph G a path of length two exists from course i to course j either through S or P, then a conflict exists and an edge must be constructed in graph K between vertex i and vertex j. This corresponds to a student's desire to take course i and course j, or a professor's assignment to teach course i and course j. For example, a path exists from vertex C1 to vertex C4 through vertex P3, which causes an edge to be constructed in K between vertex K1 and K4. Another edge must be constructed between vertex K2 and K4, because a direct path exists in G from C2 to C3

through vertex S4. Continue to construct edges in graph K until all conflict edges are drawn. The completed graph is illustrated in Figure 5. The scheduling problem is now equivalent to assigning colors to the vertices of the conflict graph so that two directly connected vertices have distinct colors.

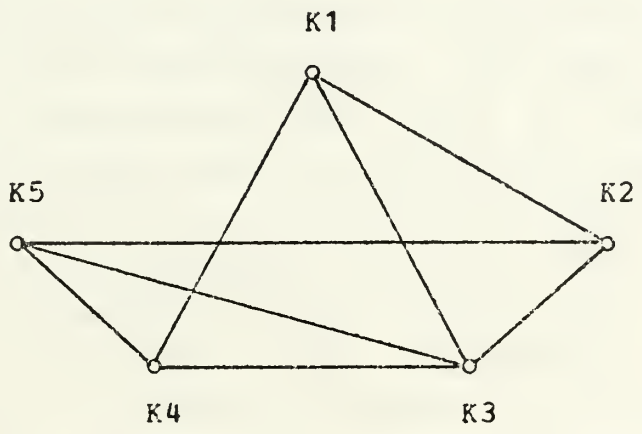

Figure 5. Conflict Graph K.

A rudimentary effort related to this field was described by Lions [2]; it involved the Queensway Senior Public School in Etobicoke, Ontario. A requirements matrix was devised and schedules were divided into weekly and then daily problems. A sample timetable for a single subgroup would be produced by one computer run and then verified by an acceptability test. The process would be repeated until all timetables were acceptable. In practice this approach is similar to random coloring of the set of vertices of K starting with any vertex and coloring it, then finding any vertex not directly connected and coloring it the same color. In Figure 5 if vertex K1 was selected to start and was assigned color one, then the only other vertex not directly connected would be vertex K5. K1 and K5 could be assigned the same color indicating course one (M-101) and course five (E-207) could be taught at the same time without

producing a conflict. The procedure tries to find another vertex which is not connected to K1 and K5. When no such vertices exist, a new color is selected and the procedure is repeated. This method is time consuming and in general will not produce an optimum schedule or a minimum number of colors.

A refinement is the Welsh-Powell algorithm for coloring the vertices of the conflict graph. The first step in the algorithm sorts the vertices into a left to right ordered list so that a vertex which has more edges connected to it occurs to the left of a vertex with less edges. The leftmost vertex is colored with color one. The first vertex which is not connected to the vertex just colored is assigned the same color. The first vertex which is not connected to all previously colored vertices is then assigned the same color. This process continues until no vertices with the above property remain. All colored vertices are deleted from the list. The leftmost vertex which remains on the list is then colored with the next color and the process is repeated until all vertices are colored. The Welsh-Powell algorithm would assign color one to vertex K3, color two to vertices K1 and K5, and color three to vertices K2 and K4 in Figure 5. Although this algorithm is efficient and will always produce a coloring, there is no guarantee that the solution will contain a minimum number of colors, which is usually desireable. However, Welsh and Powell [3] have introduced a method of determining the upper and lower bounds for the number of colors required (chromatic number).

Wood [4] proposed a slightly better method of coloring, tailored specifically to school timetable problems. A "similarity matrix" is introduced which is based on the conflict matrix K and essentially groups the vertices of K by the number of other vertices which will be affected by

coloring two distinct vertices. An algorithm is then used to produce the coloring from scanning the similarity matrix. Comparison between the similarity method and the Welsh-Powell method shows little difference for small sets of vertices but favors the similarity matrix for large numbers of vertices. An interesting effect is that the Welsh-Powell algorithm still yields a better solution when the probability of conflict is low.

D. FIXED SCHEDULE METHOD

The most successful computer implementations of school timetables are based on an entirely different mathematical concept. The course offerings are fixed in advance, based on anticipated needs and resource assets. An algorithm then assigns instructors and students to the courses in such a manner that an assignment that produces a conflict can never occur. The example problem could be solved by the same technique by first assigning times to classes and then assigning students to those times. No student may be assigned to a class which will create a conflict. If in Figure 2, class one (M-101) and class five (E-207) were offered at 0800, classes two (H-203) and four (S-105) at 0900 and class three (S-211) at 1000 then all student's choices could be satisfied. However, if classes three and four are offered at the same time then student S3 has a conflict. It becomes clear that some students may be restricted from certain combinations of classes and the preassignment of course times is critical.

This classic method is described by Clark and Stewart at the University of Maryland [5]. Basically, students filled in registration forms and a blank course offering was constructed. The computer batch processed the student's

requests assigning a student to class and blanking out the student's time vector for that time. If the vector was already blank then no assignment could be made for that period. If the course was not available at another time, an alternate was processed. As classes became filled, they were deleted. Instructor preference and other requirements such as lunch period were programmed to yield an acceptable schedule from the university point of view. Some students had requested impossible combinations so no solution existed. It is clear that the first students to be processed produced easy schedules while the latter students yielded more difficult schedules as courses filled. Order of processing did make a difference, although not a significant one.

A similar approach by Busam [6] performs in essentially the same way except that section preference is obtained by an ordering number which represents the number of students in excess of course capacity. A combinatorial approach is used while student requests are batch processed to find the first feasible solution (if one exists). Evaluation at Washington State University demonstrated the feasibility of keying on student preference but also indicated that extremely skewed requests could significantly decrease effectiveness.

E. ANALYTICAL SOLUTION

The flight scheduling problem was first approached from the graph coloring point of view. Some additional assumptions such as additional dummy aircraft assets and no preassignment were used to make this problem conform to the class scheduling problem. Two sets of sample daily data were processed using the Welsh-Powell algorithm. By

grouping aircraft which must fly as a section or division into one course and by assigning the same instructor to sorties which could not be flown at the same time, the conflict graph was reduced to a sparse graph of only thirty-three vertices. It was apparent that the method of assigning students to flights and instructors to flights played a significant role in the appearance of the conflict graph. Reversing the assignment of two instructors out of a hundred could change as many as fifteen conflict edges.

In both cases a minimal coloring of four colors (cycles) was required and obtained. In one instance, night flights were found in three of the four cycles, which proved unacceptable. Both cases yielded extremely lopsided coverings with twice as many sorties in one launch as another. In addition, there was no positive control over sortie selection. Numerous aesthetic requirements (such as offering all tactics flights on one cycle and all weapons flights on another) were violated by the resultant schedules. Thus, the graph coloring technique was rejected for several reasons. First, it became obvious that the conflict matrix did not contain sufficient information to generate a satisfactory schedule and no method to further amplify this information was apparent. Secondly, no successful method could be applied to assign students or instructors to flights which yielded better conflict graphs. Lastly, positive control over the schedule was sacrificed by calculating the entire days schedule through one analytical solution.

For the flight scheduling problem to be solved by the Fixed Schedule Method, the critical step was determining which flights would be offered during which launches. Empirical data from ninety-eight flight schedules was statistically analyzed for recurring trends with inconclusive results. Obviously, a "standard launch" was

nonexistent. Naturally, high probabilities of some sorties appearing on certain launches indicated a partial launch. It is anticipated that a careful analysis of past schedules and current scheduling technique along with the prompting priorities should yield a situation dependent "standard launch". In an effort to solve this problem and still retain sufficient human control, it was decided a temporary solution would be to have the scheduler interactively input the sortie offering. Now the scheduling problem was a duplicate of the Fixed Schedule Method.

Since students normally fly no more than once each day and must fly the appropriate sortie, the task of assigning students to sorties became trivial. Given a particular sortie, only a minimal number of students were eligible for that sortie and assignments could be made by assigning a heuristic number associated with their class and individual priority. Instructors presented an entirely different set of parameters. Highly qualified instructors, in particular the instructor RIO's (since the ratio of instructor RIO's to student pilots was low), had to be scheduled at least twice daily, which created significantly more conflicts. Clearly some method of holding the most qualified instructors in reserve until actually required would substantially reduce if not eliminate conflicts.

The solution procedure selected to solve the flight scheduling problem is based on the so called "linear assignment" problem. The solution to the assignment problem assures that men are assigned to jobs so that the total cost is minimized or a performance measure maximized. The solution uses an algorithm based on the works of Silver [7] and Munkres [8] modified by Bourgeois and Lassalle [9] to handle a rectangular cost matrix. This assignment algorithm uses a cost matrix A (i,j) which is an n X m matrix where n is the number of sorties and m is the number of instructors.

Each element represents the cost of assigning instructor j to sortie i. The cost is heuristically evaluated at twice the phase number and then adjusted for priority wing or for low flight time by subtracting appropriate weights. Nonavailability or lesser qualified instructors are handled by the "Big M" method [10] in using an exhorbitant cost to preclude assignment.

Instructor assignments are then directly dependent on the cost function assigned and may be adjusted by function modification to reflect changing parameters. Thus, the flight schedule problem has been reduced to a mathematical assignment problem for which a convenient analytical solution already exists.

IV. DESCRIPTION OF THE COMPUTER PROGRAM

A. GENERAL

The computer program was debugged and tested using sample data from past flight schedules. A particular time would be selected at random and all variables would be initialized to their value at that time. A schedule would be produced using the computer program and the final solution would be analyzed for content. In all cases, only instructors qualified to teach a sortie were assigned to that sortie. In addition, the least qualified instructor eligible was selected. Preferred wings were favored in the proposed cycles while ineligible wings were not scheduled. Since there are any number of correct solutions to a given problem, the computer generated flight schedules did not match the actual flight schedules. It was determined that the computer generated flight schedules were satisfactory because no conflicts existed nor were aesthetic requirements violated.

An example flight scheduling problem is presented to demonstrate the program's operation. Data is based on flight schedules produced during late November 1974. Student syllabus status was used to determine class composition and priority. Previously scheduled instructor events and times were used to construct instructor qualifications and wings. The VF-121 syllabus was inserted as a card file. Assets such as targets, adversary aircraft,

and the Air Combat Maneuvering Range were assumed available. Mission requirements were adjusted to accomodate aircraft availability and to permit night field carrier landing practice. While this reconstruction is not totally accurate, it does yield a sufficiently realistic environment for demonstration.

B. FILES

Extensive information is required to develop the schedule. This program simulates interactive files by using punched cards. Four major sources of information are involved.

1. STUDENTS - Current necessary information in the student file is name, class number, last date flown, category and syllabus status. This information will have to be expanded to provide improved management data.

2. INSTRUCTORS - Necessary information for scheduling instructors includes name, wing, lecture and syllabus qualification status, IUT state and date last flown. Additional information can be stored to make the system more responsive.

3. SYLLABUS - The entire syllabus must be listed chronologically. For each sortie necessary information must include the usual estimated time enroute, briefing time requirements, mission name, radar requirement code and ordinance configuration.

4. AUXILIARY - Several small files are included in this group. A coded list of positive control routes enables quicker interaction by just typing the code. Standard notes can be indexed for easy

reference. Completed schedules must be maintained to facilitate updating the files the following day. Programmed nonavailability in the form of watch bills and schools should also be maintained.

C. INPUT AND INITIALIZATION

Since the heart of the solution involves mathematical manipulation of tables or arrays, the input phase makes local table copies of important file elements. In the present form that consists of reading card files. Conversion to either seqential or random access files should not require excessive coding, since the files are copied prior to program execution. The files were punched on cards and submitted to the program as input data.

The initialization phase interactively seeks specific information about the proposed schedule such as date, duty officers and sunrise or sunset times. Many of the questions in the demonstration program could be eliminated by establishing the appropriate auxiliary files. The interactive queries and responses (simulated by card input) for the example program are listed below.

```
COMMENCE SCHEDULING PROCESS

INSERT DATE OF PROPOSED SCHEDULE.
Tuesday 20 May 1975

ENTER SCHEDULED CDO.
McDonald

ENTER SCHEDULED DAY ODO.
Arwood
```

ENTER SCHEDULED NIGHT ODO.
Volker

ENTER SCHEDULED SDO.
Ens Smith

ENTER SCHEDULED CQ ODO.
Hughes

ENTER SUNRISE TIME.
0638

ENTER SUNSET TIME.
1853

D. LIST NEXT EVENTS

The next events phase searches the student syllabus data and produces a list of the next two sorties, lectures and trainers for each student. This provides the scheduler with a recurring file check. From the sample problem output on the following page, next event data for all student pilots and RIO's is available. In case of computer failure, these compact listings are all that are required to reconstruct the student file and initiate manual scheduling. It is recommended that these daily listings be retained.

NEXT EVENT LIST

STUDENT PILOTS

NAME	SORTIES		TRAINERS		LECTURES	
PETERSON	PF-4	PF-5	SDL-1	PEW-1	FS-5	FS-6
JARONYK	PF-2	PF-3	PFW-1		FS-4	FS-5
VANDENBERT	PS-2	PS-3	PSW-5	SDL-1	FS-5	FS-6
TODD	PI-3	PS-1	SDL-1	PEW-1	FS-5	FS-6
MORAN	PF-7	PI-3	SDL-1	PEW-1	FS-6	SEP-1
CROUCH	PS-1	PS-2	PEW-1		FS-5	FS-6
HOLM	PI-4	PW-1	PEW-1		WEP-15	WEP-16
DOUGHTERTY	PW-5	PW-6	PEW-1		WEP-15	WEP-16
FINK	PW-4	PW-5	PEW-1		WEP-15	WEP-16
PHANEUF	PW-5	PW-6	PEW-1		WEP-15	WEP-16
MCCARTY	PW-5	PW-6	PEW-1		WEP-15	WEP-16
GASKELL	PW-5	PW-6	PEW-1		WEP-15	WEP-16
COOK	PW-5	PW-6	PEW-1		WEP-15	WEP-16
WILLIAMS	PT-7	PT-8	PEW-1		CQS-1	CQS-2
HOUSTEN	PT-6	PT-7	PEW-1		CQS-1	CQS-2
MULLER	PT-9	PT-10	PEW-1		CQS-1	CQS-2
WHITE	PT-9	PT-10	PEW-1		CQS-1	CQS-2
SHAFFER	PT-9	PT-10	PEW-1		CQS-1	CQS-2
CURRY	PT-11	PT-12	PEW-1		CQS-1	CQS-2
BEARD	FMLP		PEW-1		EW-2	EW-3
BERTSCH	FMLP		PEW-1		EW-2	EW-3
BLAKE	FMLP		PEW-1		EW-2	EW-3
BROWN	FMLP		PEW-1		EW-2	EW-3
BEAN	FMLP		PEW-1		EW-2	EW-3

STUDENT RIOS

NAME	SORTIES		TRAINERS		LECTURES	
TALLENT	NS-5	NS-6	NST-12	NST-15	SEP-2	SEP-3
RILEY	NS-5	NS-6	NST-12	NST-15	SEP-2	SEP-3
HEINRICH	NS-6	NS-7	NST-12	NST-15	SEP-1	SEP-2
JACOB	NS-6	NS-7	NST-12	NST-15	SEP-2	SEP-3
BOYD	NS-5	NS-6	NST-12	NST-15	SEP-2	SEP-3
MEYER	NS-5	NS-6	NST-12	NST-15	SEP-1	SEP-2
PALMER	NW-3	NW-4	NST-15	NSW-7	TAC-5	TAC-6
SAMPLE	NW-6	NW-7	NSW-7	NST-18	TAC-5	TAC-6
THOMAS	NW-5	NW-6	NST-18	NSW-8	TAC-4	TAC-5
ALLISON	NW-5	NW-6	NSW-7	NST-18	TAC-5	TAC-6
COMSTOCK	NS-13	NS-14	NST-15	NSW-7	TAC-5	TAC-6
PARKER, P	NW-3	NW-4	NSW-7	NST-18	TAC-5	TAC-6
NICHOLS	NW-4	NW-5	NST-18	NSW-8	TAC-5	TAC-6
PARKER, J	NT-7	NT-8	NST-18	NSW-8	CQS-4	
CRUMLEY	NT-8	NT-9	NST-18	NSW-8	CQS-4	
STEWART	NT-8	NT-9	NST-18	NSW-8	CQS-4	
SALGLE	NT-8	NT-9	NEW-1	NSW-8	CQS-4	
MARTIN	NT-8	NT-9	NEW-1	NSW-8	CQS-2	
PAZIK	NT-7	NT-8	NEW-1	NSW-8	CQS-4	
MENOENHALL	NT-8	NT-9	NEW-1	NSW-8	CQS-2	
OEVEER	NT-8		NEW-1	NSW-8	EW-5	
CRENSHAW	NT-8		NEW-1	NSW-8	EW-5	
NIMMER	NT-8		NEW-1	NSW-8	EW-5	
KELLNER	NT-8		NEW-1	NSW-8	EW-5	
MORRIS	NT-8		NEW-1	NSW-8	EW-5	

E. SCHEDULING PROBLEM

The scheduling problem is tenatively solved for each cycle. The sortie offering and student assignments are interactively input (Event number and launch time are included.). A situation dependent "standard launch" should automate this step. The program searches for sorties which need instructor pilots assigned, computes individual costs and then applies the assignment solution algorithm to match instructor pilots. The same process is repeated to produce instructor RIO assignments. Note that the demonstration program does have the capability to schedule instructors to trainers and lectures while solving individual cycle launches, by substituting the appropriate trainer or lecture in place of the mission name. The proposed cycle is then displayed and an edit feature is invoked to allow the scheduler to modify the cycle (if desired). The rest of the cycle launches are solved in the same manner up to an arbitrary maximum number of cycles which is currently set to four. The output and responses for the four cycles of the example schedule follow. Data for cycle four was selected to demonstrate the full manual input method for a sample LSO input for field carrier landing practice.

```
ENTER NUMBER OF SORTIES TO BE FLOWN THIS CYCLE.
00012

ENTER NUMBER OF TRAINERS OR LECTURES THIS CYCLE.
00000

ENTER MISSION-TYPE AND STUDENT FOR EACH EVENT.
MISS = NS-5      PIL =              RO = TALLENT
MISS = NS-5      PIL =              RO = RILEY
MISS = PT-7      PIL = WILLIAMS     RO =
MISS = NT-7      PIL =              RO = PARKER, J
MISS = PW-4      PIL = FINK         RO =
MISS = NW-4      PIL =              RO = PALMER
MISS = NW-7      PIL =              RO = SAMPLE
MISS = PW-5      PIL = PHANEUF      RO =
MISS = PW-6      PIL = MCCARTY      RO =
MISS = NW-6      PIL =              RO = THOMAS
MISS = PI-3      PIL = TODD         RO =
MISS = PF-7      PIL = MORAN        RO =

ENTER DESIRED INSTRUCTOR WING FOR THIS CYCLE.
00003

ENTER INELIGIBLE INSTRUCTOR WING THIS CYCLE.
00002

DO YOU DESIRE TO CHANGE PROPOSED CYCLE?
NO
                PROPOSED SCHEDULE FOR CYCLE NUMBER 1

SORTIE NUMBER        TYPE          PILOT              RIO

      01             NS-5          SHULTZ             TALLENT
      02             NS-5          THORBURN           RILEY
      03             PT-7          WILLIAMS           POWERS
      04             NT-7          SHIELDS            PARKER, J
      05             PW-4          FINK               HALMARK
      06             NW-4          PUNCHES            PALMER
      07             NW-7          HODGE              SAMPLE
      08             PW-5          PHANEUF            SMITH, P
      09             PW-6          MCCARTY            SLINERY
      10             NW-6          WATTS              THOMAS
      11             PI-3          TODD               NIMMER
      12             PF-7          MORAN              NICHOLS, M
```

```
ENTER NUMBER OF SORTIES TO BE FLOWN THIS CYCLE.
00013

ENTER NUMBER OF TRAINERS OR LECTURES THIS CYCLE.
00000

ENTER MISSION-TYPE AND STUDENT FOR EACH EVENT.
MISS = NS-6      PIL =                RO = HEINRICH
MISS = PS-1      PIL = CROUCH         RO =
MISS = PW-5      PIL = DOUGHERTY      RO =
MISS = NW-5      PIL =                RO = ALLISON
MISS = PW-5      PIL = COOK           RO =
MISS = PW-5      PIL = GASKELL        RO =
MISS = NW-5      PIL =                RO = NICHOLS
MISS = NW-3      PIL =                RO = PARKER, P
MISS = NT-8      PIL =                RO = CRUMLEY
MISS = PT-6      PIL = HOUSTEN        RO =
MISS = NT-8      PIL =                RO = STEWART
MISS = NT-8      PIL =                RO = SALGLE
MISS = PMC       PIL = COR HOUSTON    RO = SMITH, L

ENTER DESIRED INSTRUCTOR WING FOR THIS CYCLE.
00004

ENTER INELIGIBLE INSTRUCTOR WING THIS CYCLE.
00002

DO YOU DESIRE TO CHANGE PROPOSED CYCLE?
NO
```

PROPOSED SCHEDULE FOR CYCLE NUMBER 2

SORTIE NUMBER	TYPE	PILOT	RIO
01	NS-6	MARR	HEINRICH
02	PS-1	CROUCH	CRENSHAW
03	PW-5	DOUGHERTY	HALMARK
04	NW-5	HODGE	ALLISON
05	PW-5	COOK	HIERS
06	PW-5	GASKELL	TIMMESTER
07	NW-5	PUNCHES	NICHOLS
08	NW-3	HEATH	PARKER, P
09	NT-8	ANDERSON	CRUMLEY
10	PT-6	HOUSTEN	WEBB
11	NT-8	WEIGAND	STEWART
12	NT-8	HELM	SALGLE
13	PMC	COR HOUSTON	SMITH, L

```
ENTER NUMBER OF SORTIES TO BE FLOWN THIS CYCLE.
00008

ENTER NUMBER OF TRAINERS OR LECTURES THIS CYCLE.
00000

ENTER MISSION-TYPE AND STUDENT FOR EACH EVENT.
MISS = NT-8      PIL =                RO = MARTIN
MISS = NT-8      PIL =                RO = PAZIK
MISS = PT-9      PIL = MULLER         RO =
MISS = NT-8      PIL =                RO = DEVEER
MISS = PF-4      PIL = PETERSON       RO =
MISS = PF-2      PIL = JAPONYK        RO = COR SCHROEDE
MISS = PS-2      PIL = VANDENBERT     RO =
MISS = PI-1      PIL = HOLM           RO =

ENTER DESIRED INSTRUCTOR WING FOR THIS CYCLE.
00001

ENTER INELIGIBLE INSTRUCTOR WING THIS CYCLE.
00000

DO YOU DESIRE TO CHANGE PROPOSED CYCLE?
NO
```

PROPOSED SCHEDULE FOR CYCLE NUMBER 3

SORTIE NUMBER	TYPE	PILOT	RIO
01	NT-8	MCDONALD	MARTIN
02	NT-8	TERRILL	PAZIK
03	PT-9	MULLER	BJCHANEN
04	NT-8	ANDERSON	DEVEER
05	PF-4	PETERSON	WILLIAMS
06	PF-2	JARONYK	COR SCHROEDE
07	PS-2	VANDENBERT	AMBERSLY
08	PI-1	HOLM	NIMMER

```
ENTER NUMBER OF SORTIES TO BE FLOWN THIS CYCLE.
00005

ENTER NUMBER OF TRAINERS OR LECTURES THIS CYCLE.
00000

ENTER MISSION-TYPE AND STUDENT FOR EACH EVENT.
MISS = FMLP      PIL = BEARD        RO = CRENSHAW
MISS = FMLP      PIL = BERTSCH      RO = NIMMER
MISS = FMLP      PIL = BLAKE        RO = KELLNER
MISS = FMLP      PIL = BROWN        RO = MORRIS
MISS = FMLP      PIL = BEAN         RO = MENDENHALL

ENTER DESIRED INSTRUCTOR WING FOR THIS CYCLE.
00002

ENTER INELIGIBLE INSTRUCTOR WING THIS CYCLE.
00003

DO YOU DESIRE TO CHANGE PROPOSED CYCLE?
NO

                PROPOSED SCHEDULE FOR CYCLE NUMBER 4

SORTIE NUMBER          TYPE          PILOT              RIO

      01               FMLP          BEARD              CRENSHAW
      02               FMLP          BERTSCH            NIMMER
      03               FMLP          BLAKE              KELLNER
      04               FMLP          BROWN              MORRIS
      05               FMLP          BEAN               MENDENHALL
```

F. PRODUCTION RUN

The entire schedule is then displayed and edited allowing for last minute changes and to insure no errors or conflicts exist. The completed schedule may then be submitted to the Operations Officer prior to the production run which produces all desired copies on the printer. The demonstration program requires about ten seconds of computer execution time to produce the list of next events, proposed cycle schedules and to print the resultant schedule, given all interactive responses simulated by card input. The completed example flight schedule is on the following page.

FIGHTING SQUADRON ONE HUNDRED TWENTY-ONE "FLIGHT SCHEDULE" TUESDAY 20 MAY 1975

CDO: MCDONALD ODO: ARWOOD VOLKER SDO: ENS SMITH CQ ODO: HUGHES SUNRISE: 0538 SUNSET: 1853

EVT	APC#	APC ROUTE	ETD	ETE	BRF	PILOT	RIO	MISSION	A/C	ATE	C/I	S/S	REMARKS	RDR	PRI	IRD
1A	NJ25	TANGO 02 W291	0700	1.6	0515	SHULTZ	TALLENT	NS-5	...	...	...	...		R	2	P,S
1B	NJ26	TANGO 02 W291	0700	1.6	0515	THORBURN	RILEY	NS-5	...	...	...	...		R	2	P,S
2	SW25	TANGO 02 SACM	0710	1.0	0525	WILLIAMS	POWERS	PT-7	...	...	...	...	1 V1 A4E TG	R	1	P,S
3	SW26	TANGO 02 SACM	0710	1.0	0525	SHIELDS	PARKER, J	NT-7	...	...	...	...	1 V 1 T38 TG	P	1	P,S
4A	OJ25	ALPHA 24	0720	1.3	0535	FINK	HALMARK	PW-4	...	...	...	...	TOT TGT95	N	3	6 4K76
4B	OJ26	ALPHA 24	0720	1.3	0535	PUNCHES	PALMER	NW-4	...	...	...	...	0800-0830	N	3	6 4K76
4C	OJ27	ALPHA 24	0720	1.3	0535	HOOGE	SAMPLE	NW-7	...	...	...	...	284.3	N	3	6 4K76
5A	OJ28	ALPHA 24	0740	1.3	0555	PHANEUF	SMITH, P	PW-5	...	...	...	...		N	2	6 4K76
5B	OJ29	ALPHA 24	0740	1.3	0555	MCCARTY	SLINERY	PW-6	...	...	...	...		N	2	6 4K76
5C	OJ30	ALPHA 24	0740	1.3	0555	WATTS	THOMAS	NW-6	...	...	...	...		N	2	6 4K76
6	SW28	ALPHA 25	0800	1.6	0600	TODD	NIMMER	PI-3	...	...	...	...		N	2	--
7	SW29	TANGO 02 W291	0810	1.6	0625	MORAN	NICHOLS, M	PF-7	...	...	...	...		N	1	--
8A	NJ27	TANGO 02 W291	1100	1.6	0915	MARR	HEINRICH	NS-6	...	...	...	...		R	?	P,S
8B	NJ28	TANGO 02 W291	1100	1.6	0900	CROUCH	CRENSHAW	PS-1	...	...	...	...		P	?	P,S
9A	OJ31	ALPHA 24	1130	1.3	0945	DOUGHERTY	HALMARK	PW-5	...	...	...	...	TOT TGT 95	N	1	6 4K76
9B	OJ32	ALPHA 24	1130	1.3	0945	HOOGE	ALLISON	NW-5	...	...	...	...	1200-1230	N	1	6 4K76
9C	OJ33	ALPHA 24	1130	1.3	0945	COOK	HIERS	PW-5	...	...	...	...	284.3	N	1	6 4K76
10A	OJ34	ALPHA 24	1200	1.3	1015	GASKELL	TIMMESTER	PW-5	...	...	...	...	TOT TGT 95	N	2	6 4K76
10B	OJ35	ALPHA 24	1230	1.3	1045	PUNCHES	NICHOLS	NW-5	...	...	...	...	1300	N	2	6 4K76
10C	OJ36	ALPHA 24	1200	1.3	1015	HEATH	PARKER, P	NW-3	...	...	...	...	284.3	N	2	6 4K76
11A	SW30	ALPHA 25	1210	1.0	1025	ANDERSON	CRUMLEY	NT-8	...	...	...	...	2 V 1 A4E	R	1	P,S
11B	SW31	ALPHA 25	1210	1.0	1025	HOUSTEN	WEBB	PT-6	...	...	...	...	TOPGUN	R	1	P,S
12A	SW32	TANGO 02 NACM	1220	1.0	1035	WEIGAND	STEWART	NT-8	...	...	...	...	ACMR RNG	R	1	P,S
12B	SW33	TANGO 02 NACM	1220	1.0	1035	HELM	SALGLE	NT-8	...	...	...	...	1245-1315	R	1	P,S
13	SW34	TANGO 02 W291	1230	1.0	1035	COR HOUSTON	SMITH, L	PMC	...	...	...	...		R	1	P,S
14A	SW35	ALPHA 25	1600	1.0	1415	MCDONALD	MARTIN	NT-8	...	...	...	...		R	2	P,S
14B	SW36	ALPHA 25	1600	1.0	1415	TERRILL	PAZIK	NT-8	...	...	...	...		R	2	P,S
15A	SW37	ALPHA 25	1630	1.0	1430	MULLER	BUCHANEN	PT-9	...	...	...	...	2 V 1 T38	R	1	P,S
15B	SW38	ALPHA 25	1630	1.0	1445	ANDERSON	DEVEER	NT-8	...	...	...	...	TOPGUN	R	1	P,S
16	SW39	TANGO 02 W291	1640	1.6	1440	PETERSON	WILLIAMS	PF-4	...	...	...	...		N	1	--
17	SW40	TANGO 02 W291	1650	1.6	1450	JARONYK	COR SCHROEDE	PF-2	...	...	...	...		N	1	--
18	SW41	TANGO 02 W291	1700	1.6	1515	VANOENBERT	AMBERSLY	PS-2	...	...	...	...		R	2	P,S
19		DO 175	1710	1.5	1510	HOLM	NIMMER	PI-1	...	...	...	...		N	1	--
20A			2150	0.5	2050	BEARD	CRENSHAW	FMLP	...	...	...	...	NKX "C"	N	1	--
20B			2150	0.5	2050	BERTSCH	NIMMER	FMLP	...	...	...	...	2200 7.5	N	1	--
20C			2150	0.5	2050	BLAKE	KELLNER	FMLP	...	...	...	...	FUEL REQD	N	1	--
20D			2150	0.5	2050	BROWN	MORRIS	FMLP	...	...	...	...	LSO: MORSE	N	1	--
20E			2150	0.5	2050	BEAN	MENDENHALL	FMLP	...	...	...	...		N	1	--

V. PROPOSED EXTENSION TO AN INTERACTIVE SYSTEM

A. STATUS

It must be emphasized that the coded demonstration program is not the complete system, but it can be modified to perform all the required functions. The main concept of this system is interactive where constant user action is required (positive control). The demonstration program was developed using COBOL to enable conversion for fleet use. This necessitated batch operation to be run on the IBM 360/67 at the Church Computer Center, Naval Postgraduate School, because the COBOL compiler is not available under the time sharing operating system. This batch simulation of an interactive program precluded full software development.

B. HARDWARE ASSETS REQUIRED

Effective utilization of the proposed system requires some computer hardware not normally present in squadrons but often available through remote government computer facilities. The general characteristics are summarized.

1. COMPUTER TIME - Sufficient computer time must be available on a time sharing basis during anticipated run times for COBOL operation. Core required for program run step is anticipated to be less than 100K exclusive of file management and

compiler.

2. CATHODE RAY TERMINAL - Although a high speed teletypewriter could be used, a CRT is more effective for file management, display and editing purposes. Standard telephone communications lines should be sufficient for data transmission.

3. STORAGE - Permanent file storage, preferably on disk (although magnetic tape could be used), must be available at the computer facility. Although the files are not classified, some privacy should be afforded and access limited to certain users.

4. CARD READER - The current program is on punched cards. Initial loading and debugging as well as file creation requires access to a card reader.

5. REMOTE PRINTER - The printer must be locally stationed to enable easy retrieval of printed output in the form of dumped files and, most importantly, multiple copies of the finished flight schedule. A slow speed upper case printer is acceptable.

C. SOFTWARE REQUIRED

Prior to computing the new flight schedule, the files must be updated to reflect the results of the previous day's training. A Data Base Management System is required to provide the user with a simple method of file creation and updating prior to program execution.

Since no interactive terminal was available for the demonstration program, the interactive input and output will require the most modification to convert to a suitable

format for Cathode Ray Terminal (CRT) operation. This area is most responsive to user desires and changes and should be developed under closer liason. Including file manipulation, time sharing and user "think" time, it is anticipated that the program should not require more than forty-five minutes total per run.

VI. SUMMARY

A. LIMITATIONS

Conflict graph coloring techniques provided a convenient means of analyzing the flight scheduling problem but were of limited utility in this application. School timetable problems contained the same essential variables and published solutions to these problems provided valuable insight to the flight scheduling problem. Existence of an adequate assignment algorithm considerably simplified this work but the final algorithm had to be translated from ALGOL to COBOL.

Since this computer program was the author's first COBOL programming effort, a top down modular programming technique was employed to enable easier development. Absence of an interactive COBOL CRT was a major drawback in system development. Another major limitation was offsite system analysis which caused general unavailability of quick answers.

B. SYSTEM UTILITY

Although the proposed system was designed as a solution to the scheduling problem at VF-121, the mathematical concepts are sufficienty broad to enable application to

other problems. For the bi-partite student and instructor relationship of flight schedules for this training command, the solution developed in this thesis should prove acceptable. However, this technique may not be applicable to operational squadrons. Should analysis reveal that conflicts are not related to instructor assignments, then another solution, such as the one suggested by Boeck [11], should be explored.

It is anticipated that full implementation of the proposed interactive system can be accomplished using the part time services of a COBOL programmer and close liason with the scheduling staff over a three month period. Adequate test results should be available within a month and a half to base an intelligent acceptance or rejection decision. Other commands desiring to use this system should anticipate longer implementation times to allow for individual analysis of the scheduling problem.

C. CONCLUSIONS

Successful implementation of this system in a time sharing environment should result in significant personnel savings by reducing the scheduling staff to one officer. The files can serve as a basis for improved management decisions and later expansion could result in larger savings in this area. It has been demonstrated that a satisfactory schedule can be produced by using the computer. The solution procedure does not sacrifice positive control by the user. The computer thus becomes a powerful tool which increases the efficiency of the flight scheduling operation.

```
IDENTIFICATION DIVISION.
PROGRAM-ID.  SCHEDULE.
AUTHOR.  LT CRAIG G HONOUR.
INSTALLATION.  NAVAL POSTGRADUATE SCHOOL CHURCH CENTER.
DATE-WRITTEN.  MAY 1975.
SECURITY.  UNCLASSIFIED.


ENVIRONMENT DIVISION.
CONFIGURATION SECTION.


SOURCE-COMPUTER. IBM-360.
OBJECT-COMPUTER. IBM-360.

INPUT-OUTPUT SECTION.
FILE-CONTROL.
     SELECT STUDIN ASSIGN TO UR-S-IN1.
     SELECT ACCPT ASSIGN TO UR-S-IN2.
     SELECT SOR-IN ASSIGN TO UR-S-IN3.
     SELECT SKEDOUT ASSIGN TO UR-S-OUT1.

DATA DIVISION.
FILE SECTION.
FD   STUDIN
     LABEL RECORDS ARE OMITTED
     DATA RECORD IS NAME-CARD.

01   NAMECARD.
     32   PERSON          PIC A(12).
     02   FILLER          PIC XX.
     02   NUM             PIC 999.
     02   FILLER          PIC XX.
     02   WING            PIC 9.
     02   FILLER          PIC XX.
     02   LQ              PIC 999.
     02   FILLER          PIC XX.
     02   SQ              PIC 999.
     02   FILLER          PIC XX.
     02   IUT             PIC 999.
     02   FILLER          PIC XX.
     02   LED             PIC 999.
     02   FILLER          PIC X(40).

FD   ACCPT
     LABEL RECORDS ARE OMITTED
     DATA RECORD IS AC-CARD.
```

```
01  AC-CARD.
    02  DIG            PIC 9(5).
    02  MSG            PIC X(25).
    02  FILLER         PIC X(50).

FD  SOR-IN
    LABEL RECORDS ARE OMITTED
    DATA RECORD IS SOR-CARD, CYCLE-CARD.

01  SOR-CARD.
    02  MISSI          PIC X(8).
    02  FILLER         PIC XX.
    02  ETEI           PIC 9V9.
    02  FILLER         PIC XXX.
    02  BRFI           PIC 9(4).
    02  FILLER         PIC X.
    02  RDRI           PIC A.
    02  FILLER         PIC XXXX.
    02  ORDI           PIC X(14).
    02  FILLER         PIC X.
    02  INSTI          PIC A(12).
    02  FILLER         PIC X(28).

01  CYCLE-CARD.
    02  PIL            PIC A(12).
    02  FILLER         PIC XX.
    02  RO             PIC A(12).
    02  FILLER         PIC XXX.
    02  MISS           PIC X(8).
    02  FILLER         PIC XX.
    02  SNUM           PIC 99.
    02  FILLER         PIC X.
    02  IEVT           PIC X(3).
    02  FILLER         PIC XX.
    02  IAPC           PIC 9.
    02  FILLER         PIC X.
    02  IETD           PIC 9999.
    02  FILLER         PIC X.
    02  IREM           PIC X(12).
    02  FILLER         PIC X.
    02  IPRI           PIC 9.
    02  FILLER         PIC X.
    02  IAPCN          PIC X(4).
    02  FILLER         PIC X(7).

FD  SKEDOUT
    LABEL RECORDS ARE OMITTED
    DATA RECORD IS SKED-LINE.
```

```
01  SKED-LINE.
    02 FILLER    PIC X.
    02 EVT-O     PIC XXX.
    02 FILLER    PIC X.
    02 APC-O     PIC XXXX.
    02 FILLER    PIC XX.
    02 APCR-O    PIC X(13).
    02 FILLER    PIC XX.
    02 ETD-O     PIC 9(4).
    02 FILLER    PIC XX.
    02 ETE-O     PIC 9.9.
    02 FILLER    PIC XX.
    02 BRF-O     PIC 9(4).
    02 FILLER    PIC XXX.
    02 PILOT-O   PIC A(12).
    02 FILLER    PIC XX.
    02 RIO-O     PIC A(12).
    02 FILLER    PIC XX.
    02 MISSION-O PIC X(8).
    02 FILLER     PIC XXX.
    02 DOTS       PIC X(15).
    02 FILLER     PIC XXX.
    02 REMARK-O   PIC X(12).
    02 FILLER     PIC XXX.
    02 RDR-O     PIC A.
    02 FILLER     PIC XXX.
    02 PRI-O      PIC 9.
    02 FILLER     PIC XX.
    02 ORD-O      PIC X(14).

WORKING-STORAGE SECTION.
77  N                  PIC 99       VALUE 30.
77  M                  PIC 99       VALUE 50.
77  MIN                PIC S99      COMP.
77  COL                PIC S9999,   COMP.
77  CL                 PIC S9999,   COMP.
77  CLO                PIC S9999,   COMP.
77  CL1                PIC S9999,   COMP.
77  I                  PIC S9999,   COMP, SYNC.
77  J                  PIC S9999,   COMP.
77  K                  PIC S9999,   COMP.
77  L                  PIC S9999,   COMP.
77  RL                 PIC S9999,   COMP.
77  RS                 PIC S9999,   COMP.
77  IMIN               PIC S9999,   COMP.
```

```
77   IMAX                PIC S9999,  COMP.
77   FLAG                PIC S9999,  COMP.
77   SW                  PIC S9999,  COMP.
77   Z1                  PIC S9999,  COMP.
77   Z2                  PIC S9999,  COMP.
77   P1                  PIC 999     VALUE 51.
77   P2                  PIC 999     VALUE 101.
77   P3                  PIC 999     VALUE 161.
77   P4                  PIC 999     VALUE 181.
77   P5                  PIC 999     VALUE 215.
77   LASTDATE            PIC 999     VALUE 218.
77   DOTS1               PIC X(15)   VALUE '... ... ... ...'.
77   COD                 PIC 9.
77   CYCLE-NUM           PIC 9.
77   PRIOR               PIC 9.
77   PRIOR1              PIC 9.
77   K1                  PIC 99.
77   OLD                 PIC 99.
77   N1                  PIC 99.
77   SORTIE-NUM          PIC 99.
77   TOT                 PIC 99.
77   M1                  PIC 99.
77   Z3                  PIC 999.
77   TOTAL               PIC 999.


01   TOP-LINE.
     02  FILLER          PIC X(18)   VALUE SPACE.
     02  STMT            PIC X(35)
                         VALUE 'PROPOSED SCHEDULE FOR CYCLE NUMBER '.
     02  C-N             PIC 9.

01   COL-HEAD1.
     02  FILLER          PIC X       VALUE SPACE.
     02  S-COL           PIC X(13)   VALUE 'SORTIE NUMBER'.
     02  FILLER          PIC X(10)   VALUE SPACES.
     02  T-COL           PIC X(4)    VALUE 'TYPE'.
     02  FILLER          PIC X(10)   VALUE SPACES.
     02  P-COL           PIC X(5)    VALUE 'PILOT'.
     02  FILLER          PIC X(17)   VALUE SPACES.
     02  RIO-COL         PIC X(3)    VALUE 'RIO'.
     02  FILLER          PIC X(70)   VALUE SPACES.

01   SKED-LINE1.
     02  FILLER          PIC X(7)    VALUE SPACES.
     02  S-N             PIC 99.
     02  FILLER          PIC X(15)   VALUE SPACES.
```

```
    02  TY              PIC X(8).
    02  FILLER          PIC X(6)     VALUE SPACES.
    02  P-NAM           PIC X(12).
    02  FILLER          PIC X(10)    VALUE SPACES.
    02  RIO-NAM         PIC X(12).
    02  FILLER          PIC X(61)    VALUE SPACES.


01  SKED-HEAD1.
    02 FILLER           PIC X        VALUE SPACES.
    02 SQUADRON         PIC X(40)
          VALUE 'FIGHTING SQUADRON ONE HUNDRED TWENTY-ONE'.
    02 FILLER           PIC X(20)    VALUE SPACES.
    02 F-S              PIC X(17)    VALUE '"FLIGHT SCHEDULE"'.
    02 FILLER           PIC X(21)    VALUE SPACES.
    02 DAT              PIC X(25).

01  SKED-HEAD2.
    02 FILLER           PIC X        VALUE SPACES.
    02 CDO              PIC X(6)     VALUE 'CDO:  '.
    02 CDO-NAME         PIC X(12).
    02 FILLER           PIC XXX      VALUE SPACES.
    02 ODO              PIC X(6)     VALUE  'ODO:  '.
    02 ODO-NAM1         PIC X(12).
    02 FILLER           PIC X        VALUE SPACE.
    02 ODO-NAM2         PIC X(12).
    02 FILLER           PIC XXX      VALUE SPACES.
    02 SDO              PIC X(6)     VALUE 'SDO:  '.
    02 SDO-NAME         PIC X(12).
    02 FILLER           PIC XXX      VALUE SPACES.
    02 CQODO            PIC X(9)     VALUE 'CQ ODO:  '.
    02 CQ-NAME          PIC X(12).
    02 FILLER           PIC XXX      VALUE SPACES.
    02 SUNRISE          PIC X(10)    VALUE 'SUNRISE:  '.
    02 SUNR-TIME        PIC X(4).
    02 FILLER           PIC X(3)     VALUE SPACES.
    02 SUNSET           PIC  X(9)    VALUE 'SUNSET:  '.
    02 SUNS-TIME        PIC X(4).
    02 FILLER           PIC X(16)    VALUE SPACES.

01  COL-HEAD.
    02 FILLER           PIC X        VALUE SPACES.
    02 AA               PIC XXX      VALUE 'EVT'.
    02 FILLER           PIC X        VALUE SPACES.
    02 B                PIC XXXX     VALUE 'APC#'.
    02 FILLER           PIC XX       VALUE SPACES.
    02 CC               PIC X(9)     VALUE 'APC ROUTE'.
```

```
    02 FILLER          PIC X(6)     VALUE SPACES.
    02 D               PIC XXX      VALUE 'ETD'.
    02 FILLER          PIC XXX      VALUE SPACES.
    02 E               PIC XXX      VALUE 'ETE'.
    02 FILLER          PIC XX       VALUE SPACES.
    02 F               PIC XXX      VALUE 'BRF'.
    02 FILLER          PIC XXXX     VALUE SPACES.
    02 G               PIC X(5)     VALUE 'PILOT'.
    02 FILLER          PIC X(9)     VALUE SPACES.
    02 H               PIC X(3)     VALUE 'RIO'.
    02 FILLER          PIC X(11)    VALUE SPACES.
    02 II              PIC X(11)    VALUE 'MISSION  '.
    02 I2              PIC X(8)     VALUE 'A/C ATE '.
    02 JJ              PIC X(7)     VALUE 'C/I G/S'.
    02 FILLER          PIC X(3)     VALUE SPACES.
    02 JJJ             PIC X(7)     VALUE 'REMARKS'.
    02 FILLER          PIC X(5)     VALUE SPACES.
    02 KK              PIC X(13)    VALUE '  RDR PRI ORD'.
    02 FILLER          PIC X(18)    VALUE SPACES.

01  EVENT-HEAD.
    02  FILLER        PIC X(29),   VALUE SPACES.
    02  H-LINE        PIC X(15),   VALUE 'NEXT EVENT LIST'.
    02  FILLER        PIC X(89),   VALUE SPACES.

01  PILOT-HEAD.
    02  FILLER        PIC X(30),   VALUE SPACES.
    02  P-HEAD        PIC X(14),   VALUE 'STUDENT PILOTS'.
    02  FILLER        PIC X(88),   VALUE SPACES.

01  RIO-HEAD.
    02  FILLER        PIC X(30),   VALUE SPACES.
    02  R-HEAD        PIC X(12),   VALUE 'STUDENT RIOS'.
    02  FILLER        PIC X(90),   VALUE SPACES.

01  EVT-COL-HEAD.
    02  FILLER        PIC X,       VALUE SPACE.
    02  E-NAM         PIC X(4),    VALUE 'NAME'.
    02  FILLER        PIC X(16),   VALUE SPACES.
    02  E-SOR         PIC X(7),    VALUE 'SORTIES'.
    02  FILLER        PIC X(11),   VALUE SPACES.
    02  E-TRN         PIC X(8),    VALUE 'TRAINERS'.
    02  FILLER        PIC X(12),   VALUE SPACES.
    02  E-LEC         PIC X(8),    VALUE 'LECTURES'.
    02  FILLER        PIC X(64),   VALUE SPACES.

01  EVENT-LINE.
    02  FILLER        PIC X,       VALUE SPACE.
```

```
    02  EL-NAME      PIC A(12).
    02  FILLER       PIC X(5),   VALUE SPACES.
    02  SOR1         PIC X(5).
    02  FILLER       PIC X(3),   VALUE SPACES.
    02  SOR2         PIC X(5).
    02  FILLER       PIC X(5),   VALUE SPACES.
    02  TR1          PIC X(6).
    02  FILLER       PIC X(3),   VALUE SPACES.
    02  TR2          PIC X(6).
    02  FILLER       PIC X(5),   VALUE SPACES.
    02  LEC1         PIC X(6).
    02  FILLER       PIC X(5),   VALUE SPACES.
    02  LEC2         PIC X(6).
    02  FILLER       PIC X(60),  VALUE SPACES.

01  C-TABLE.
    02 C             PIC S9999,  COMP, OCCURS 30 TIMES.
01  CB-TABLE.
    02 CB            PIC S9999,  COMP, OCCURS 50 TIMES.
01  LAMBDA-TABLE.
    02 LAMBDA        PIC S9999,  COMP, OCCURS 50 TIMES.
01  MU-TABLE.
    02 MU            PIC S9999,  COMP, OCCURS 30 TIMES.
01  R-TABLE.
    02 R             PIC S9999,  COMP, OCCURS 30 TIMES.
01  Y-TABLE.
    02 Y             PIC S9999,  COMP, OCCURS 50 TIMES.


01  A-TABLE.
    02 A-ROW                     OCCURS 30 TIMES.
       03  A         PIC S9999,  COMP, OCCURS 50 TIMES.
01  X-TABLE.
    02 X             PIC S99,    COMP, OCCURS 50 TIMES.

01  SYL-TABLE.
    02  SYL-ROW                 OCCURS 60 TIMES.
        03  SYL      PIC 9,      OCCURS 215 TIMES.
01  SHP-TABLE.
    02  SHP-ROW                 OCCURS 60 TIMES.
        03  SHP      PIC X(5),   OCCURS 6 TIMES.

01  MISSION-TABLE.
    02  MISSION     PIC X(8),   OCCURS 215 TIMES.
01  ETE-TABLE.
    02  ETE         PIC 9V9,    OCCURS 100 TIMES.
01  BRF-TABLE.
```

```
    02  BRF             PIC 9(4),    OCCURS 100 TIMES.
01  RDR-TABLE.
    02  RDR             PIC A,       OCCURS 100 TIMES.
01  ORD-TABLE.
    02  ORD             PIC X(14),   OCCURS 100 TIMES.
01  EVT-TABLE.

    02  EVT             PIC XXX,     OCCURS 50 TIMES.
01  APC-TABLE.
    02  APC             PIC X(4),    OCCURS 50 TIMES.
01  APCR-TABLE.
    02  APCRC           PIC 9,       OCCURS 50 TIMES.
01  ETD-TABLE.
    02  ETD             PIC 9(4),    OCCURS 50 TIMES.
01  PILOT-TABLE.
    02  PILOT           PIC A(12),   OCCURS 50 TIMES.
01  RIO-TABLE.
    02  RIO             PIC A(12),   OCCURS 50 TIMES.
01  MIS-TABLE.
    02  MIS             PIC X(8),    OCCURS 50 TIMES.
01  REM-TABLE.
    02  REM             PIC X(12),   OCCURS 50 TIMES.
01  PRI-TABLE.
    02  PRI             PIC 9,       OCCURS 50 TIMES.
01  INSTRUCT-TABLE.
    02  INSTRUCT        PIC A(12),   OCCURS 50 TIMES.
01  ROUTE-TABLE.
    02 ROUTE            PIC X(13),   OCCURS 150 TIMES.

01  STUD-TABLE.
    02  STUD                         OCCURS 60 TIMES.
        03  STUD-NAME                PIC A(12).
        03  STUD-CLASS-NUM           PIC 999.
        03  LASTFLY                  PIC 999.
        03  STUD-CAT                 PIC 9.

01  INST-TABLE.
    02  INST                         OCCURS 100 TIMES.
        03  INST-NAME                PIC A(12).
        03  INST-WING                PIC 9.
        03  LEC-QUAL                 PIC 999.
        03  SYL-QUAL                 PIC 999.
        03  IUT-STAT                 PIC 9.
        03  LASTFLOWN                PIC 999.
```

```
PROCEDURE DIVISION.
MAIN SECTION.
     OPEN INPUT STUDIN.
     OPEN INPUT ACCPT.
     OPEN INPUT SOR-IN.
     OPEN OUTPUT SKEDOUT.
     PERFORM IN-1 THRU INIXIT VARYING I FROM 1 BY 1 UNTIL I > 60.
     PERFORM IN-3 VARYING I FROM 1 BY 1 UNTIL I > 100.
     PERFORM IN4 THRU IN4XIT VARYING I FROM 1 BY 1 UNTIL I > 9.
     PERFORM IN-5.
INIT.
     DISPLAY 'COMMENCE SCHEDULING PROCESS'.
     DISPLAY '  '.
     DISPLAY '  '.
     DISPLAY 'INSERT DATE OF PROPOSED SCHEDULE.'.
     READ ACCPT AT END GO TO ERRORIN.
     DISPLAY '  '.
     EXHIBIT MSG
     MOVE MSG TO DAT.
     DISPLAY '  '.
     DISPLAY '  '.
     DISPLAY 'ENTER SCHEDULED CDO.'.
     READ ACCPT AT END GO TO ERRORIN.
     DISPLAY '  '.
     EXHIBIT MSG
     MOVE MSG TO CDO-NAME.
     DISPLAY '  '.
     DISPLAY '  '.
     DISPLAY 'ENTER SCHEDULED DAY ODO.'.
     READ ACCPT AT END GO TO ERRORIN.
     DISPLAY '  '.
     EXHIBIT MSG
     MOVE MSG TO ODO-NAM1.
     DISPLAY '  '.
     DISPLAY '  '.
     DISPLAY  'ENTER SCHEDULED NIGHT ODO.'.
     READ ACCPT AT END GO TO ERRORIN.
     DISPLAY '  '.
     EXHIBIT MSG
     MOVE MSG TO ODO-NAM2.
     DISPLAY '  '.
     DISPLAY '  '.
     DISPLAY 'ENTER SCHEDULED SDO.'.
     READ ACCPT AT END GO TO ERRORIN.
     DISPLAY '  '.
     EXHIBIT MSG
```

```
     MOVE MSG TO SDO-NAME.
     DISPLAY '  '.
     DISPLAY '  '.
     DISPLAY 'ENTER SCHEDULED CQ ODO.'.
     DISPLAY '  '.
     READ ACCPT AT END GO TO ERRORIN.
     EXHIBIT MSG
     MOVE MSG TO CQ-NAME.
     DISPLAY '  '.
     DISPLAY '  '.
     DISPLAY 'ENTER SUNRISE TIME'.
     DISPLAY '  '.
     READ ACCPT AT END GO TO ERRORIN.
     EXHIBIT MSG
     MOVE MSG TO SUNR-TIME.
     DISPLAY '  '.
     DISPLAY '  '.
     DISPLAY 'ENTER SUNSET TIME.'.
     READ ACCPT AT END GO TO ERRORIN.
     DISPLAY '  '.
     EXHIBIT MSG
     MOVE MSG TO SUNS-TIME.
     DISPLAY '  '.
     DISPLAY '  '.
     DISPLAY '  '.
     DISPLAY '  '.
     DISPLAY '  '.
     PERFORM SETHOPS THRU SETXIT.
     MOVE 0 TO Z3.
     MOVE 1 TO CYCLE-NUM.
HOP-SKED.
     PERFORM LISTHOP THRU LISTXIT.
     PERFORM FIXCOST.
     PERFORM SHCYCLE.
     PERFORM EDIT THRU EDITXIT.
     ADD 1 TO CYCLE-NUM.
     ADD TOT TO Z3.
     IF CYCLE-NUM < 5, GO TO HOP-SKED.
EOJ.
     PERFORM SHSKED.
     PERFORM EDITSKED THRU EDITSXIT.
     CLOSE SKEDOUT, SOR-IN, ACCPT, STDIN.
     STOP RUN.

ERRORIN.
     DISPLAY 'ERROR IN INPUT. PROGRAM TERMINATION.'.
     GO TO EOJ.
```

```
INDATA SECTION.

IN-1.
    READ STUDIN AT END GO TO ERRORIN.
    MOVE PERSON TO STUD-NAME (I).
    MOVE NUM TO STUD-CLASS-NUM (I).
    MOVE WING TO STUD-CAT (I).
    MOVE LFD TO LASTFLY (I).
    PERFORM IN-11 VARYING J FROM 1 BY 1 UNTIL J > P5.
    IF I > 30, GO TO IN-2.
    PERFORM IN-12 VARYING J FROM 1 BY 1 UNTIL J > SQ.
    PERFORM IN-12 VARYING J FROM P2 BY 1 UNTIL J > LQ.
    PERFORM IN-12 VARYING J FROM P3 BY 1 UNTIL J > IUT.
    GO TO INIXIT.

IN-2.
    PERFORM IN-12 VARYING J FROM P1 BY 1 UNTIL J > SQ.
    PERFORM IN-12 VARYING J FROM P2 BY 1 UNTIL J > LQ.
    PERFORM IN-12 VARYING J FROM P4 BY 1 UNTIL J > IUT.
INIXIT. EXIT.

IN-11.
    MOVE 0 TO SYL (I, J).
IN-12.
    MOVE 1 TO SYL (I, J).


IN-3.
    READ STUDIN AT END GO TO ERRORIN.
    MOVE PERSON TO INST-NAME (I).
    MOVE WING TO INST-WING (I).
    MOVE LQ TO LEC-QUAL (I).
    MOVE SQ TO SYL-QUAL (I).
    MOVE IUT TO IUT-STAT (I).
    MOVE LFD TO LASTFLOWN (I).

IN4.
    READ ACCPT AT END GO TO ERRORIN.
    MOVE MSG TO ROUTE (I).
IN4XIT.  EXIT.

IN-5.
```

```
    PERFORM READ1 VARYING I FROM 1 BY 1 UNTIL I > 100.
    PERFORM READ2 VARYING I FROM 101 BY 1 UNTIL I > 215.
READ1.
    READ SOR-IN AT END GO TO ERRORIN.
    MOVE MISSI TO MISSION (I).
    MOVE ETEI TO ETE (I).
    MOVE BRFI  TO BRF (I).
    MOVE RORI  TO ROR (I).
    MOVE ORDI TO ORD (I).
READ2.
    READ SOR-IN AT END GO TO ERRORIN.
    MOVE MISSI TO MISSION (I).


SETHOPS.
    WRITE SKED-LINE FROM EVENT-HEAD AFTER 20.
    WRITE SKED-LINE FROM PILOT-HEAD AFTER 3.
    WRITE SKED-LINE FROM EVT-COL-HEAD AFTER 3.
    MOVE SPACES TO SKED-LINE.
    WRITE SKED-LINE AFTER 1.
    MOVE 1 TO I.
    MOVE 0 TO OLD.
SET1.
    IF STUD-CLASS-NUM (I) = 0, GO TO SET2.
    MOVE STUD-NAME (I) TO EL-NAME.
    DIVIDE 100 INTO STUD-CLASS-NUM (I) GIVING TOTAL.
    IF TOTAL = OLD, GO TO SET11.
    MOVE TOTAL TO OLD.
    MOVE SPACES TO SKED-LINE.
    WRITE SKED-LINE AFTER 1.
SET11.
    MOVE 0 TO Z1, Z2.
    PERFORM LOOP1 THRU LOOP11 VARYING J FROM 1 BY 1 UNTIL
                    Z2 > 0 OR J = P1.
    IF Z1 NOT = 0, MOVE MISSION (Z1) TO SOR1, SHP (I, 1).
    IF Z2 NOT = 0, MOVE MISSION (Z2) TO SOR2, SHP (I, 2),
        ELSE MOVE SPACES TO SOR2.
    MOVE 0 TO Z1, Z2.
    PERFORM LOOP1 THRU LOOP11 VARYING J FROM P2 BY 1 UNTIL
                    Z2 > 0 OR J = P3.
    IF Z1 NOT = 0, MOVE MISSION (Z1) TO LEC1, SHP (I, 3).
    IF Z2 NOT = 0, MOVE MISSION (Z2) TO LEC2, SHP (I, 4),
        ELSE MOVE SPACES TO LEC2.
    MOVE 0 TO Z1, Z2.
    PERFORM LOOP1 THRU LOOP11 VARYING J FROM P3 BY 1 UNTIL
                    Z2 > 0 OR J = P4.
    IF Z1 NOT = 0, MOVE MISSION (Z1) TO TR1, SHP (I, 5).
```

```
    IF Z2 NOT = 0, MOVE MISSION (Z2) TO TR2, SHP (I, 6),
        ELSE MOVE SPACES TO TR2.
    ADD 1 TO I.
    WRITE SKED-LINE FROM EVENT-LINE AFTER 1.
    GO TO SET1.
SET2.
    MOVE 31 TO I.
    WRITE SKED-LINE FROM RIO-HEAD AFTER 5.
    WRITE SKED-LINE FROM EVT-COL-HEAD AFTER 3.
    MOVE SPACES TO SKED-LINE.
    WRITE SKED-LINE AFTER 1.
    MOVE 0 TO OLD.
SET21.
    IF STUD-CLASS-NUM (I) = 0, GO TO SETXIT.
    MOVE STUD-NAME (I) TO EL-NAME.
    DIVIDE 100 INTO STUD-CLASS-NUM (I) GIVING TOTAL.
    IF TOTAL = OLD, GO TO SET22.
    MOVE TOTAL TO OLD.
    MOVE SPACES TO SKED-LINE.
    WRITE SKED-LINE AFTER 1.
SET22.
    MOVE 0 TO Z1, Z2.
    PERFORM LOOP1 THRU LOOP11 VARYING J FROM P1 BY 1 UNTIL
                    Z2 > 0 OR J = P2.
    IF Z1 NOT = 0, MOVE MISSION (Z1) TO SOR1, SHP (I, 1).
    IF Z2 NOT = 0, MOVE MISSION (Z2) TO SOR2, SHP (I, 2),
        ELSE MOVE SPACES TO SOR2.
    MOVE 0 TO Z1, Z2.
    PERFORM LOOP1 THRU LOOP11 VARYING J FROM P2 BY 1 UNTIL
                    Z2 > 0 OR J = P3.
    IF Z1 NOT = 0, MOVE MISSION (Z1) TO LEC1, SHP (I, 3).
    IF Z2 NOT = 0, MOVE MISSION (Z2) TO LEC2, SHP (I, 4),
        ELSE MOVE SPACES TO LEC2.
    MOVE 0 TO Z1, Z2.
    PERFORM LOOP1 THRU LOOP11 VARYING J FROM P4 BY 1 UNTIL
                    Z2 > 0 OR J = P5.
    IF Z1 NOT = 0, MOVE MISSION (Z1) TO TR1, SHP (I, 5).
    IF Z2 NOT = 0, MOVE MISSION (Z2) TO TR2, SHP (I, 6),
        ELSE  MOVE SPACES TO LEC2.
    ADD 1 TO I.

    WRITE SKED-LINE FROM EVENT-LINE AFTER 1.
    GO TO SET21.
LOOP1.
    IF SYL (I, J) = 1, GO TO LOOP11.
    IF Z1 = 0, MOVE J TO Z1, ELSE MOVE J TO Z2.
LOOP11.  EXIT.
SETXIT.  EXIT.
```

```
FIXCOST.
    DISPLAY '  '.
    DISPLAY '  '.
    DISPLAY 'ENTER DESIRED INSTRUCTOR WING FOR THIS CYCLE.'.
    READ ACCPT AT END GO TO ERRORIN.
    DISPLAY '  '.
    EXHIBIT DIG
    MOVE DIG TO PRIOR.
    DISPLAY '  '.
    DISPLAY '  '.
    DISPLAY 'ENTER INELIGIBLE INSTRUCTOR WING THIS CYCLE.'.
    DISPLAY '  '.
    READ ACCPT AT END GO TO ERRORIN.
    EXHIBIT DIG
    DISPLAY '  '.
    DISPLAY '  '.
    MOVE DIG TO PRIOR1.
    MOVE 0 TO N1.
    PERFORM DOPILOT THRU DOXIT VARYING I FROM 1 BY 1
        UNTIL I > TOT.
    ADD 1 N1 GIVING Z1.
    PERFORM MAKEO VARYING I FROM Z1 BY 1 UNTIL I > 30
        AFTER J FROM 1 BY 1 UNTIL J > 50.
    PERFORM MATCH.
    MOVE 0 TO N1.
    PERFORM DOPILOT1 THRU DOXIT1 VARYING I FROM 1 BY 1
        UNTIL I > TOT.
    MOVE 0 TO N1.
    PERFORM DORIO THRU DORXIT VARYING I FROM 1 BY 1
        UNTIL I > TOT.
    ADD 1 N1 GIVING Z1.
    PERFORM MAKEO VARYING I FROM Z1 BY 1 UNTIL I > 30
        AFTER J FROM 1 BY 1 UNTIL J > 50.
    PERFORM MATCH.
    MOVE 0 TO N1.
    PERFORM DORIO1 THRU DORXIT1 VARYING I FROM 1 BY 1
        UNTIL I > TOT.

DOPILOT.
    ADD I Z3 GIVING J.
    IF PILOT (J) = '                    ', GO TO GETCOST.
    GO TO DOXIT.
GETCOST.
    ADD 1 TO N1.
    MOVE 1 TO CL.
LOOP3.
```

```
LIST SECTION.

LISTHOP.
     DISPLAY '  '.
     DISPLAY '  '.
     DISPLAY '  '.
     DISPLAY 'ENTER NUMBER OF SORTIES TO BE FLOWN THIS CYCLE.'.
     READ ACCPT AT END GO TO ERRORIN.
     DISPLAY '  '.
     EXHIBIT DIG
     DISPLAY '  '.
     DISPLAY '  '.
     MOVE DIG TO SORTIE-NUM.
     DISPLAY 'ENTER NUMBER OF TRAINERS OR LECTURES THIS CYCLE.'.
     DISPLAY '  '.
     READ ACCPT AT END GO TO ERRORIN.
     EXHIBIT DIG
     MOVE DIG TO K1.
     ADD K1 SORTIE-NUM GIVING TOT.
     DISPLAY '  '.
     DISPLAY '  '.
     DISPLAY 'ENTER MISSION-TYPE AND STUDENT FOR EACH EVENT.'.
     MOVE CYCLE-NUM TO C-N.
     PERFORM CYCLE VARYING J FROM 1 BY 1 UNTIL J > TOT.
     GO TO EOJ2.
CYCLE.
     READ SOR-IN AT END GO TO ERRORIN.
     ADD J 23 GIVING M1.
     EXHIBIT NAMED MISS, PIL, RO
     MOVE PIL TO P-NAM, PILOT (M1).
     MOVE RO TO RIO-NAM, RIO (M1).
     MOVE MISS TO TY, MIS (M1).
     MOVE IEVT TO EVT (M1).
     MOVE IAPC TO APCRC (M1).
     MOVE IETD TO ETD (M1).
     MOVE IREM TO REM (M1).
     MOVE IPRI TO PRI (M1).
     MOVE IAPCN TO APC (M1).
EOJ2.
LISTXIT.  EXIT.

FIX SECTION.
```

```
    IF CL < 216, GO TO LOOP4.
    DISPLAY 'NO MISSION MATCH FOUND FOR' MIS (J).
CONT2.
    MOVE 1 TO COD.
    IF CL > 10 AND CL < 20, MOVE 2 TO COD.
    IF CL > 21 AND CL < 31, MOVE 3 TO COD.
    IF CL > 31 AND CL < 46, MOVE 4 TO COD.
    PERFORM CODE2 THRU CODE2X VARYING K FROM 1 BY 1 JNTIL K > 50.
DORXIT.  EXIT.

CODE2.
    ADD K 50 GIVING COL.
    IF COD > SYL-QUAL (COL), MOVE 99 TO A (N1, K),
        ELSE MULTIPLY SYL-QUAL (COL) BY 2 GIVING A (N1, K).
    IF INST-WING (COL) = PRIOR, SUBTRACT 1 FROM A (N1, K).
CODE2X.  EXIT.

DORIO1.
    ADD I Z3 GIVING J.
    IF RIO (J) = '                ', GO TO FILL2.
    GO TO DORXIT1.

FILL2.
    ADD 1 TO N1.
    ADD 50 X (N1) GIVING Z1.
    MOVE INST-NAME (Z1) TO RIO (J).
DORXIT1.


PRINT SECTION.

SHSKED.
    WRITE SKED-LINE FROM SKED-HEAD1 AFTER 20.
    WRITE SKED-LINE FROM SKED-HEAD2 AFTER 5.
    WRITE SKED-LINE FROM COL-HEAD AFTER 3.
    MOVE SPACES TO SKED-LINE.
    WRITE SKED-LINE AFTER 1.
    PERFORM SKED1 THRU SKEDXIT VARYING I FROM 1 BY 1
        UNTIL I > Z3.


SKED1.
    MOVE 1 TO Z1.
SKED11.
    IF MISSION (Z1) = MIS (I), GO TO SKED2.
    ADD 1 TO Z1.
    IF Z1 < 100, GO TO SKED11.
    GO TO SKED3.
```

```
    IF MIS (J) = MISSION (CL), GO TO CONT1.
    ADD 1 TO CL.
    IF CL < 216, GO TO LOOP3.
    DISPLAY 'NO MISSION MATCH FOR' MIS (J).
CONT1.
    MOVE 1 TO COD.
    IF CL > 55 AND CL < 73, MOVE 2 TO COD.
    IF CL > 73 AND CL < 82, MOVE 3 TO COD.
    IF CL > 82 AND CL < 98, MOVE 4 TO COD.
    PERFORM CODE1 THRU CODXIT VARYING K FROM 1 BY 1 UNTIL K > 50.
DOXIT.  EXIT.


CODE1.
    IF COD > SYL-QUAL (K) OR PRIOR1 = INST-WING (K),
        GO TO COD11.
    MULTIPLY SYL-QUAL (K) BY 2 GIVING A (N1, K).
    IF LASTFLOWN (K) < LASTDATE, SUBTRACT 1 FROM A (N1, K).
    IF INST-WING (K) = PRIOR, SUBTRACT 1 FROM A (N1, K).
    GO TO CODXIT.
COD11.
    MOVE 99 TO A (N1, K).
CODXIT.  EXIT.

MAKE0.
    MOVE 0 TO A (I, J).

DOPILOT1.
    ADD I Z3 GIVING J.
    IF PILOT (J) = '                ', GO TO FILL1.
    GO TO DOXIT1.
FILL1.
    ADD 1 TO N1.
    MOVE X (N1) TO Z1.
    MOVE INST-NAME (Z1) TO PILOT (J).
DOXIT1.  EXIT.

DORIO.
    ADD I Z3 GIVING J.
    IF RIO (J) = '                ', GO TO GETCOST1.
    GO TO DORXIT.

GETCOST1.
    ADD 1 TO N1.
    MOVE 1 TO CL.
LOOP4.
    IF MIS (J) = MISSION (CL), GO TO CONT2.
    ADD 1 TO CL.
```

```
    DISPLAY 'ENTER NEW PILOT, RIO, MISSION AND SORTIE NUMBER.'.
    NOTE  THIS SIMULATES FILE MANAGEMENT SYSTEM.
    ADD Z3 TO SNUM.
    READ SOR-IN AT END GO TO EOJ.
    MOVE PIL TO PILOT (SNUM).
    MOVE RO TO RIO (SNUM).
    MOVE MISS TO MIS (SNUM).
    PERFORM SHSKED.
    GO TO EDIT.
EDITXIT.  EXIT.


EDITSKED.
    DISPLAY 'DO YOU DESIRE TO CHANGE THE SCHEDULE?'.
    NOTE  ACTJAL EDITING TO BE PERFORMED BY FILE MANAGEMENT
        SYSTEM ON CATHODE RAY TUBE.
EDITSXIT.  EXIT.


MATCH SECTION.


    MOVE 0 TO TOTAL.
    MOVE M TO IMIN.
    MOVE N TO IMAX.
    IF N > M, GO TO JA.
    MOVE N TO IMIN.
    MOVE M TO IMAX.
    PERFORM B1 VARYING I FROM 1 BY 1 JNTIL I > N.
    IF M > N, GO TO JB.

JA.
    PERFORM JA4 VARYING J FROM 1 BY 1 UNTIL J > M.

JB.
    PERFORM JB1 VARYING I FROM 1 BY 1 UNTIL I > N.
    PERFORM JB2 VARYING J FROM 1 BY 1 UNTIL J > M.
    PERFORM JB3 THRU J1 VARYING I FROM 1 BY 1 UNTIL I > N
            AFTER J FROM 1 BY 1 UNTIL J > M.

START1.
    COMPUTE FLAG = N.
    COMPUTE RL = 0.
    COMPUTE CL = 0.
    COMPUTE RS = 1.
    PERFORM ST1 THRU I1 VARYING I FROM 1 BY 1 JNTIL I > N.
    IF FLAG = IMIN, GO TO FINI.
    PERFORM ST2 VARYING J FROM 1 BY 1 JNTIL J > M.
```

```
SKED2.
     SUBTRACT BRF (Z1) FROM ETD (I) GIVING BRF-O.
     MOVE ETE (Z1) TO ETE-O.
     MOVE RDR (Z1) TO RDR-O.
     MOVE ORD (Z1) TO ORD-O.
SKED3.
     MOVE MIS (I) TO MISSION-O.
     MOVE EVT (I) TO EVT-O.
     MOVE APC (I) TO APC-O.
     MOVE APCRC (I) TO Z1.
     MOVE ROUTE (Z1) TO APCR-O.
     MOVE ETD (I) TO ETD-O.
     MOVE PILOT (I) TO PILOT-O.
     MOVE RIO (I) TO RIO-O.
     MOVE REM (I) TO REMARK-O.
     MOVE PRI (I) TO PRI-O.
     MOVE DOTS1 TO DOTS.
     WRITE SKED-LINE AFTER 2.
SKEDXIT.  EXIT.


SHCYCLE.
     WRITE SKED-LINE FROM TOP-LINE AFTER 20.
     WRITE SKED-LINE FROM COL-HEAD1 AFTER 4.
     MOVE SPACES TO SKED-LINE.
     WRITE SKED-LINE AFTER 3.
     PERFORM CYCLE1 VARYING J FROM 1 BY 1 UNTIL J > TOT.
CYCLE1.
     MOVE J TO S-N.
     ADD J Z3 GIVING K.
     MOVE PILOT (K) TO P-NAM.
     MOVE RIO (K) TO RIO-NAM.
     MOVE MIS (K) TO TY.
     WRITE SKED-LINE FROM SKED-LINE1 AFTER 2.


EDITS SECTION.

EDIT.
     DISPLAY 'DO YOU DESIRE TO CHANGE PROPOSED CYCLE?'.
     READ ACCPT AT END GO TO ERRORIN.
     DISPLAY '  '.
     EXHIBIT MSG
     DISPLAY '  '.
     DISPLAY '  '.
     DISPLAY '  '.
     DISPLAY '  '.
     IF MSG = 'NO', GO TO EDITXIT.
```

```
JB2.
    COMPUTE Y (J) = 0.
JB3.
    IF A (I, J) NOT = 0 OR X (I) NOT = 0, GO TO J1.
    IF Y (J) NOT = 0, GO TO J1.
    COMPUTE X (I) = J.
    COMPUTE Y (J) = I.
J1. EXIT.


ST1.
    COMPUTE MU (I) = 0.
    IF X (I) NOT = 0, GO TO I1.
    COMPUTE RL = RL + 1.
    COMPUTE R (RL) = I.
    COMPUTE MU (I) = - 1.
    COMPUTE FLAG = FLAG - 1.
I1. EXIT.

ST2.
    COMPUTE LAMBDA (J) = 0.

LAB1.
    IF A (I, J) NOT = 0 OR LAMBDA (J) NOT = 0, GO TO J2.
    COMPUTE LAMBDA (J) = I.
    COMPUTE CL = CL + 1.
    COMPUTE C (CL) = J.
    IF Y (J) = 0, GO TO MARK.
    COMPUTE RL = RL + 1.
    MOVE Y (J) TO Z1.
    COMPUTE R (RL) = Z1.
    COMPUTE MU (Z1) = I.
J2. EXIT.

LAB2.
    IF LAMBDA (J) NOT = 0, GO TO J3.
    COMPUTE COL = COL + 1.
    COMPUTE CB (COL) = J.
J3. EXIT.

LAB3.
    MOVE R (K) TO Z1.
    MOVE CB (L) TO Z2.
    IF A (Z1, Z2) < MIN, COMPUTE MIN = A (Z1, Z2).
LAB31. EXIT.

LAB4.
    IF MU (I) NOT = 0, GO TO I2.
```

```
LABL.
    COMPUTE I = R (RS).
    COMPUTE RS = RS + 1.
    PERFORM LAB1 THRU J2 VARYING J FROM 1 BY 1 UNTIL J > M.
    IF RS NOT > RL, GO TO LABL.
    COMPUTE SW = 1.
    COMPUTE CLO = CL.
    COMPUTE COL = 0.
    PERFORM LAB2 THRU J3 VARYING J FROM 1 BY 1 UNTIL J > M.
    MOVE R (I) TO Z1.
    MOVE CB (I) TO Z2.
    COMPUTE MIN = A (Z1, Z2).
    PERFORM LAB3 THRU LAB31 VARYING K FROM 1 BY 1 UNTIL K > RL
            AFTER L FROM 1 BY 1 UNTIL L > COL.
    COMPUTE TOTAL = TOTAL + MIN *  (RL + COL - IMAX).
    PERFORM LAB4 THRU I3 VARYING I FROM 1 BY 1 UNTIL I > N.
    IF SW = 3, GO TO NEXT1.
    IF SW = 4, GO TO MARK.

NEXT1.
    IF CLO = CL, GO TO LABL.
    COMPUTE CL1 = CLO + 1.
    PERFORM NEXT11 VARYING I FROM CL1 BY 1 UNTIL I > CL.
    GO TO LABL.

SUB-SECTION.

B1.
    MOVE A (I, 1) TO MIN.
    PERFORM JA2 THRU JA21 VARYING J FROM 2 BY 1 UNTIL J > M.
    PERFORM JA3 VARYING J FROM 1 BY 1 UNTIL J > M.
    COMPUTE TOTAL = TOTAL + MIN.

JA2.
    IF A (I, J) < MIN,  MOVE A (I, J) TO MIN.
JA21. EXIT.
JA3.
    COMPUTE A (I, J) = A (I, J) - MIN.
JA4.
    COMPUTE MIN = A (I, J).
    PERFORM JA2 THRU JA21 VARYING I FROM 2 BY 1 UNTIL I > N.
    PERFORM JA3 VARYING I FROM 1 BY 1 UNTIL I > N.
    COMPUTE TOTAL = TOTAL + MIN.


JB1.
    COMPUTE  X (I) = 0.
```

```
    IF CLO < 1,  GO TO I3.
    PERFORM A1 VARYING L FROM 1 BY 1 JNTIL L > CLO.
    GO TO I3.
I2.
    PERFORM A2 THRU L1 VARYING L FROM 1 BY 1 UNTIL L > COL.
I3. EXIT.


A1.
    MOVE C (L) TO Z1.
    COMPUTE A (I, Z1) = A (I, Z1) + MIN.
A2.
    MOVE CB (L) TO Z1.
    COMPUTE A (I, Z1) = A (I, Z1) - MIN.
    IF SW = 1, GO TO NEXT2.
    IF SW = 2, GO TO L1.
    IF SW = 3, GO TO NEXT1.
    IF SW = 4, GO TO MARK.
NEXT2.
    MOVE CB (L) TO Z1.
    IF A (I, Z1) NOT = 0 OR LAMBDA (Z1) NOT = 0, GO TO L1.
    COMPUTE LAMBDA (Z1) = I.
    IF Y (Z1) = 0, PERFORM NEXT21.
    COMPUTE CL = CL + 1.
    COMPUTE C (CL) = CB (L).
    COMPUTE RL = RL + 1.
    MOVE CB (L) TO Z1.
    COMPUTE R (RL) = Y (Z1).
L1. EXIT.

NEXT21.
    COMPUTE J = CB (L).
    COMPUTE SW = 2.
    GO TO L1.
NEXT11.
    MOVE C (I) TO Z1.
    MOVE Y (Z1) TO Z2.
    COMPUTE MU (Z2) = Z1.
MARK1.
    COMPUTE X (I) = J.
    GO TO START1.
MARK.
    COMPUTE Y (J) = LAMBDA (J).
    COMPUTE I = LAMBDA (J).
    IF X (I) = 0, GO TO MARK1.
    COMPUTE K = J.
    COMPUTE J = X (I).
```

```
    COMPUTE X (I) = K.
    GO TO MARK.
FINI. EXIT.
```

LIST OF REFERENCES

1. Busacker, Robert G., and Saaty, Thomas L., Finite Graphs and Networks: An Introduction With Applications, p. 65, McGraw-Hill, Inc., 1965.

2. Lions, J., " The Ontario School Scheduling Problem", Computer Journal, v. 10, p. 14-21, May 1967.

3. Welsh, D. and Powell, M., " An Upper Bound for the Chromatic Number of a Graph and its Application to Timetabling Problems", Computer Journal, v. 10, p. 85, May 1967.

4. Wood D., " A Technique for Coloring a Graph Applicable to Large Scale Timetabling Problems", Computer Journal, v. 12, p. 317-319, November 1969.

5. Stewart, James and Clark, Robert, " University of Maryland Student Scheduling Algorithm", Proceedings of ACM 23 National Conference, v. 1, p. 555-562, 1968.

6. Busam, Vincent A., " An Algorithm for Class Scheduling With Section Preference", Communications of the ACM, v. 10, p. 567-569, September 1967.

7. Silver, R., " Algorithm for the Assignment Problem", Communications of the ACM, v. 3, p. 605-606, November 1960.

8. Munkres, J., " Algorithm for the Assignment and Transportation Problems", Journal SIAM, v. 5, p. 32-38, March 1957.

9. Bourgeois, F. and Lassalle, J.C., " Algorithm for the Assignment Problem (Rectangular Matrices)", Communications of the ACM, v. 14, p. 805-806, December 1971.

10. Hillier, Fredrick S. and Lieberman, Gerald, Introduction to Operations Research, p. 63, Holder-Day Inc., 1974.

11. Boeck, Walter G., An Interactive and Heuristic Computer System for Military Flight Training Scheduling, Masters Thesis, University of Hawaii, November 1972.

INITIAL DISTRIBUTION LIST

		No. Copies
1.	Defense Documentation Center Cameron Station Alexandria, Virginia 22314	2
2.	Library, Code 0212 Naval Postgraduate School Monterey, California 93940	2
3.	Chairman, Code 72 Computer Science Group Naval Postgraduate School Monterey, California 93940	1
4.	Professor Uno Kodres, Code 72Kr Naval Postgraduate School Monterey, California 93940	1
5.	LT Craig G. Honour, USN 385A Bergin Dr. Monterey, California 93940	1
6.	Commanding Officer Fighter Squadron One Hundred Twenty-one Naval Air Station Miramar San Diego, California 92145	1